**22**

**Phân tích số liệu dãy thời gian**

**(Time Series Analysis)**

Một trong những dữ liệu phổ biến trong kinh tế và y khoa là dữ liệu mang tính thời gian và chu kì. Chẳng hạn như tỉ lệ tử vong theo năm, tỉ giá đồng USD được ghi chép từng ngày qua nhiều năm, nhịp tim của bệnh nhân được máy ghi nhận theo từng phút trong một ngày. Trong thuật ngữ tiếng Anh, những dữ liệu này được đề cập đến như là “time series data” – dữ liệu theo thời gian. Những dữ liệu theo thời gian có đặc điểm là chúng biến chuyển theo chu kì và có xu hướng. Nói cách khác, “phía dưới” những biến chuyển của dữ liệu theo thời gian là một số yếu tố mà phân tích thống kê cần phải phát hiện.

Phương pháp dữ liệu theo thời gian bao gồm các phương pháp “chiết xuất” các chỉ số thống kê phản ảnh những đặc điểm của dữ liệu. Một hướng phân tích khác là xây dựng mô hình tiên lượng dữ liệu tương lai dựa vào những dữ liệu đã quan sát trong quá khứ. Dĩ nhiên, các mô hình hồi qui tuyến tính cũng có thể sử dụng cho tiên lượng, nhưng các mô hình như thế có vài vấn đề về giả định và khó ứng dụng cho dữ liệu theo thời gian. Ngoài một số package chuyên dụng như **TSA, tseries**, và **timeSeries**, trong R đã có sẵn một số hàm cơ bản để phân tích các dữ liệu theo thời gian. Trong chương này, tôi sẽ giới thiệu phương pháp đơn giản và một số hàm cơ bản cho phân tích dữ liệu. Các phương pháp đó bao gồm phân tích mô tả xu hướng và chu kì (hay mùa), và các mô hình Moving Average, Autoregressive, và ARIMA.

**22.1 Dữ liệu và nhập dữ liệu**

Việc cần làm đầu tiên trong phân tích dữ liệu theo thời gian là nhập dữ liệu theo đúng thời gian. Trong chương này chúng ta sẽ làm quen với 3 dữ liệu theo thời gian.

**Ví dụ 1: Tuổi tử vong của vua Anh Quốc**. Số liệu dưới đây phản ảnh tuổi tử vong của 42 vị vua của Anh Quốc. Trong trường hợp này, dữ liệu không hẳn mang tính thời gian, vì khoảng cách giữa các năm tử vong của vua không bằng nhau. Thông thường dữ liệu chỉ là một dãy số liệu như sau:

60 43 67 50 56 42 50 65 68 43 65 34 47 34 49 41 13 35 53 56 16 43 69 59 48 59 86 55 68 51 33 49 67 77 81 67 71 81 68 70 77 56

**Ví dụ 2: Bán hàng lưu niệm.** Số liệu thu thập về lượng hàng lưu niệm bán cho du khách ở một vùng du lịch thuộc bang Queensland (Úc) từ 1/1987 đến 12/1993:

1664.81 2397.53 2840.71 3547.29 3752.96

3714.74 4349.61 3566.34 5021.82 6423.48

7600.60 19756.21 2499.81 5198.24 7225.14

4806.03 5900.88 4951.34 ...

Đây là dữ liệu theo thời gian vì có tháng và năm. Chúng ta có thể “đọc” dữ liệu vào R với hàm scan như sau:

souv = scan("~/Documents/\_Vietnam14/R Book (Revised)/Data/souvenirs.txt")

Nhưng dữ liệu souv chưa thể phân tích được vì chỉ là một vector số mà không có yếu tố tháng và năm. Để thêm yếu tố tháng và năm, chúng ta cần dùng đến hàm ts như sau:

ts.souv = ts(souv, frequency=12, start=c(1987, 1))

Bây giờ dữ liêu ts.souv đã có một hình thức mới và có thể sử dụng cho phân tích:

> ts.souv

Jan Feb Mar Apr May Jun

1987 1664.81 2397.53 2840.71 3547.29 3752.96 3714.74

1988 2499.81 5198.24 7225.14 4806.03 5900.88 4951.34

1989 4717.02 5702.63 9957.58 5304.78 6492.43 6630.80

1990 5921.10 5814.58 12421.25 6369.77 7609.12 7224.75

1991 4826.64 6470.23 9638.77 8821.17 8722.37 10209.48

1992 7615.03 9849.69 14558.40 11587.33 9332.56 13082.09

1993 10243.24 11266.88 21826.84 17357.33 15997.79 18601.53

...

**Ví dụ 3: Số trẻ em mới sinh ở New York.** Dữ liệu dưới đây là số trẻ em mới sinh mỗi tháng ở Thành phố New York từ 1/1946 đến 12/1959 cũng là một dữ liệu theo thời gian:

26.663 23.598 26.931 24.740 25.806 24.364 24.477 23.901 23.175

23.227 21.672 21.870 21.439 21.089 23.709 21.669 21.752 20.761

23.479 23.824 23.105 23.110 21.759 22.073 21.937 20.035 23.590

21.672 22.222 22.123 23.950 23.504 22.238 23.142 21.059 21.573

21.548 20.000 22.424 20.615 21.761 22.874 24.104 23.748 23.262

22.907 21.519 22.025 22.604 20.894 24.677 23.673 25.320 23.583

...

Mặc dù dữ liệu không có cột tháng và năm, nhưng chúng ta có thể dùng hàm R để tạo ra một dữ liệu R (data frame) cho phân tích. Trước hết, đọc dữ liệu vào đối tượng births.

births=scan("~/Documents/\_Vietnam14/\_R Book (Revised)/Data/births.txt")

Sau đó, sắp xếp dữ liệu theo từng tháng, bắt đầu từ 1/1946 đến 12/1959, bằng cách dùng hàm ts trong R, và cho kết quả vào đối tượng ts.births:

ts.births = ts(births, frequency=12, start=c(1946, 1))

Trong hàm trên, chú ý lệnh frequency=12 (một năm có 12 tháng) và start=c(1946, 1) tức bắt đầu từ tháng 1 năm 1946. Kiểm tra kết quả cho thấy R đã sắp xếp dữ liệu theo tháng năm:

> ts.births

Jan Feb Mar Apr May Jun Jul Aug Sep

1946 26.663 23.598 26.931 24.740 25.806 24.364 24.477 23.901 23.175

1947 21.439 21.089 23.709 21.669 21.752 20.761 23.479 23.824 23.105

1948 21.937 20.035 23.590 21.672 22.222 22.123 23.950 23.504 22.238

1949 21.548 20.000 22.424 20.615 21.761 22.874 24.104 23.748 23.262

1950 22.604 20.894 24.677 23.673 25.320 23.583 24.671 24.454 24.122

...

**22.2 Biểu đồ**

Sau khi dữ liệu đã được sắp xếp và nhập thích hợp cho phân tích, bước đầu tiên là xem xét dữ liệu bằng một biểu đồ. Hàm **plot.ts** có thể giúp cho phân tích này. Nên nhớ đối tượng của **plot.ts** là dữ liệu đã được hàm **ts** nhập:

plot.ts (ts.births)



**Biểu đồ số trẻ mới sinh mỗi tháng ở New York từ 1/1946 đến 12/1959**

Có thể thấy từ biểu đồ này một chu kì mang tính mùa và xu hướng năm. Trong mỗi năm số trẻ mới sinh thường cao nhất vào mùa hè và giảm thấp nhất vào mùa đông. Nhưng đồng thời xu hướng chung là tính từ 1950 thì số trẻ mới sinh tăng theo năm.

Đối với dữ liệu về tuổi tử vong của các vua Anh Quốc, chúng ta có biểu đồ dưới đây:

kings = scan("~/Documents/\_Vietnam14/R Book (Revised)/Data/kings.txt")

ts.kings = ts(kings)

plot.ts(ts.kings)



**Biểu đồ mô tả tuổi tử vong của 42 vua Anh Quốc**

Biểu đồ trên cho thấy không có một chu kì rõ ràng, tuy nhiên có xu hướng. Sự biến chuyển của dữ liệu có thể mô tả bằng một mô hình cộng hưởng (additive model), vì độ dao động ngẫu nhiên có vẻ bất biến (độc lập) với thời gian.

Nhưng biểu đồ mô tả biến chuyển về lượng hàng lưu niệm bán ra thì có xu hướng và chu kì rõ ràng:

souv=scan("~/Documents/\_Vietnam14/\_R Book (Revised)/Data/souvenirs.txt")

ts.souv = ts(souv, frequency=12, start=c(1987, 1))

plot.ts(ts.souv)



**Biểu đồ mô tả lượng hàng lưu niệm bán ra từ 1/1987 đến 12/1993.**

Biểu đồ cho thấy mô hình cộng hưởng có vẻ không thích hợp để mô tả dữ liệu này, vì có bằng chứng cho thấy có chu kì trong dữ liệu. Có lẽ chúng ta cần hoán chuyển dữ liệu sang đơn vị log để xem mô hình cộng hưởng có thể dùng để mô tả:

ts.souv = ts(souv, frequency=12, start=c(1987, 1))

plot.ts(ts.souv)

logsouv = log(ts.souv)

plot.ts(logsouv)



**Biểu đồ mô tả lượng hàng lưu niệm (đơn vị log) bán ra từ 1/1987 đến 12/1993.**

Biểu đồ trên cho thấy mức độ dao động theo chu kì có vẻ độc lập với thời gian Do đó, có lẽ chúng ta sẽ dùng đơn vị log cho các phân tích kế tiếp.

**22.3 Phân tích các thành phần trong dữ liệu theo thời gian**

Dãy số liệu theo thời gian có 3 thành phần (components) chính theo thuật ngữ tiếng Anh như sau: *trend*, *seasonal*, và *cyclic*. Ba thành phần này có thể giải thích như sau:

**Trend (xu hướng lâu dài)**

Xu hướng là chiều hướng biến đổi, tăng hay giảm, trong dữ liệu. Xu hướng không nhất thiết phải là tuyến tính.

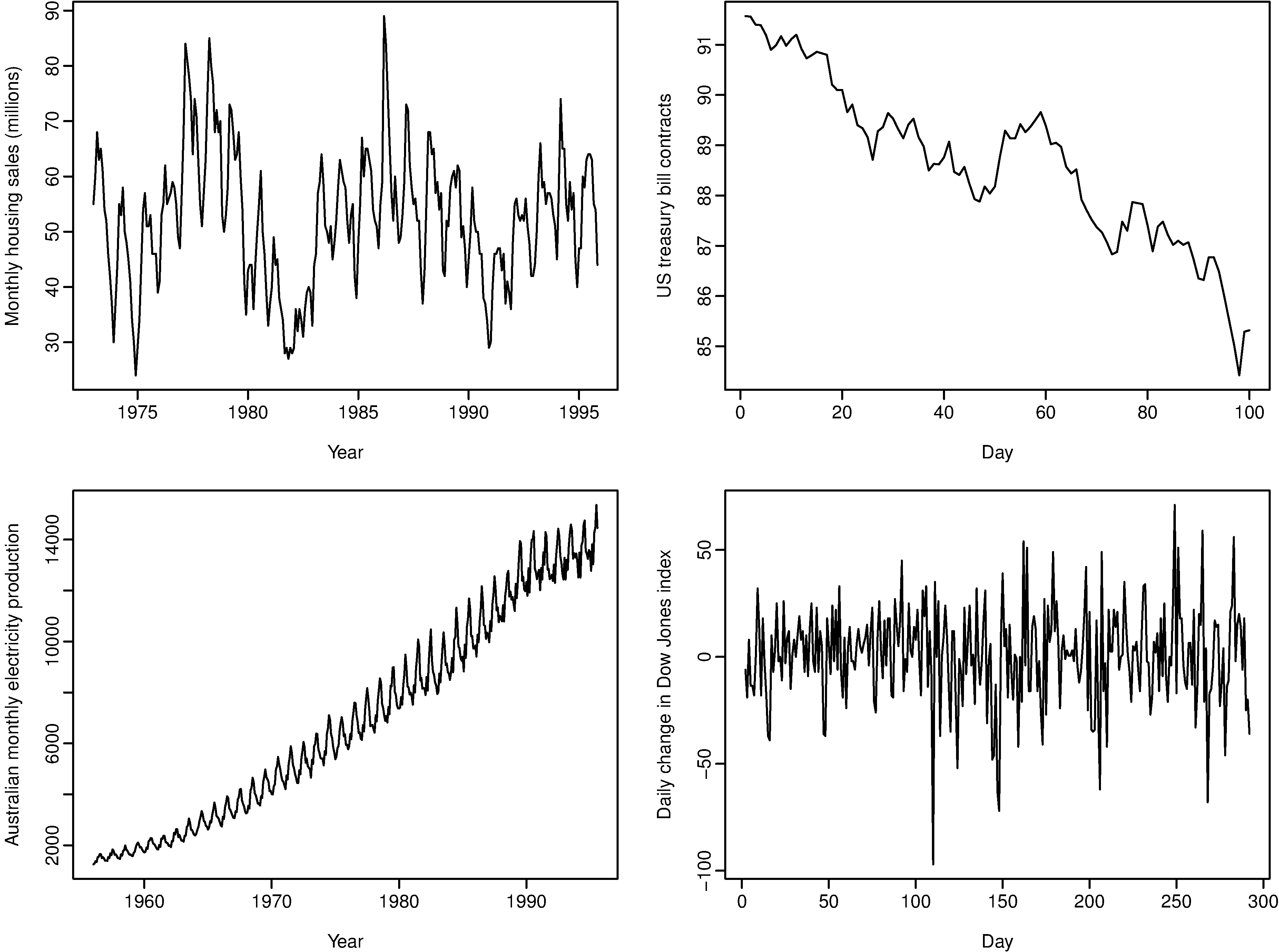
**Seasonal (theo mùa)**

Biến đổi theo mùa xảy ra khi dãy số liệu chịu ảnh hưởng bởi các yếu tố mùa (như quí trong năm, tháng, hay ngày trong tuần). Biến đổi theo mùa lúc nào cũng xảy ra trong một quãng thời gian nhất định.

**Cyclic (chu kì)**

Biến đổi theo chu kì tồn tại khi dữ liệu thể hiện những biến chuyển (tăng và giảm) mà *không theo một thời gian cố định*. Nhiều dữ liệu theo thời gian dao động tăng giảm trong thời gian tối thiểu là 2 năm.

Nếu người nhầm lẫn giữa biến chuyển cyclic và seasonal, nhưng thật ra sự phân biệt không quá khó khăn. Nếu dao động không xảy ra trong một thời gian cố định, đó là biến chuyển cyclic; nếu thời gian cố định (như tuần, tháng, quí) biến chuyển là seasonal. Nói chung, thời gian của biến chuyển cyclic thường dài hơn thời gian của biến chuyển seasonal. Ngoài ra, bức độ của biến chuyển cyclic cũng thường cao hơn biến chuyển seasonal.



Bốn loại dữ liệu theo thời gian. **Biểu đồ phía trái dòng 1** cho thấy biến đổi theo mùa trong mỗi năm, nhưng cũng có biến đổi cyclic trong thời gian 6-10 năm. Tuy nhiên dữ liệu không biểu hiện xu hướng lâu dài. **Biểu đồ bên phải dòng 1** không biểu hiện biến chuyển theo mùa, nhưng xu hướng giảm thì rất rõ ràng. **Biểu đồ phía trái dòng 2** biểu hiện xu hướng tăng rất rõ ràng, với thành phần seasonal cũng rất rõ ràng. Tuy nhiên, không có biểu hiện cyclic. **Biểu đồ bên phải dòng 2** không có xu hướng lâu dài, không có yếu tố seasonal, và cũng không có biến đổi cyclic. Tất cả có vẻ chỉ là ngẫu nhiên, rất khó tiên lượng.

Cần phân biệt 2 loại dữ liệu theo thời gian: loại không có yếu tố seasonal và loại có yếu tố seasonal.

### 22.3.1 Đối với dữ liệu không seasonal

Đối với dữ liệu không có season (như tuổi thọ của vua Anh Quốc) dữ liệu chỉ có 2 phần: phần hệ thống (regular component) và phần phi hệ thống hay ngẫu nhiên (irrgular component). Gọi dãy dữ liệu là *Y,* để tách hai phần này, chúng ta cần phải ước tính phần xu hướng (*Tt*), và phần còn lại là *Et:*

*Yt* = *Tt* + *Et*

Để ước tính *T* của dữ liệu không có season, chúng ta có thể dùng một phương pháp *mượt hoá* (gọi là *smoothing method*). Một trong những phương pháp mượt hoá phổ biến là ***simple moving average*** (SMA), tôi tạm dịch là *trung bình động đơn giản*.

**Trung bình động** là một số trung bình của một tập hợp số liệu được tính toán *tích luỹ* theo thời gian và *trong một khoản thời gian* nhất định. Đây là một chỉ số thống kê rất có ích cho việc xác định xu hướng của một dãy số liệu theo thời gian. Ví dụ chúng ta có một dãy số liệu theo năm như sau:

2003: 4 triệu

2004: 6 "

2005: 5 "

2006: 8 "

2007: 9 "

2008: 5 "

2009: 4 "

2010: 3 "

2011: 7 "

Chúng ta có thể chọn tính trung bình động cho một khoản thời gian. Trung bình động 3 năm có thể tính như sau:

(4 + 6 + 5) / 3 = 5

cho 3 năm kế tiếp: (6 + 5 + 8)/3 = 6.3

và 3 năm kế tiếp: (5 + 8 + 9)/3 = 7.3

vân vân. Tính như thế chúng ta có một dãy số mới chỉ toàn là trung bình động.

Tương tự, chúng ta cũng có thể tính trung bình động cho 5 năm thay vì 3 năm. Trung bình động 5 năm như sau:

(4 + 6 + 5 + 8 + 9) / 5 = 6.4

(6 + 5 + 8 + 9 + 5) / 5 = 6.6

(5 + 8 + 9 + 5 + 4) / 5 = 6.2

v.v.

Chúng ta có thêm 2 biến như sau:

|  |  |  |  |
| --- | --- | --- | --- |
| **Năm** | **Số liệu** | **SMA=3** | **SMA=5** |
| 2003 | 4 |  |  |
| 2004 | 6 |  |  |
| 2005 | 5 | 5.0 |  |
| 2006 | 8 | 6.3 |  |
| 2007 | 9 | 7.3 | 6.4 |
| 2008 | 5 | 7.3 | 6.6 |
| 2009 | 4 | 6.0 | 6.2 |
| 2010 | 3 | 4.0 | 5.8 |
| 2011 | 7 | 4.7 | 5.6 |

SMA=3 là trung bình động tính trên 3 năm; SMA=5 là trung bình động tính cho 5 năm.

Như có thể xem qua bảng số liệu trên, trung bình động với thời gian càng rộng thì xu hướng càng ổn định, xu hướng càng “mượt mà”. Tuy nhiên, trung bình động với thông số *n* quãng thời gian càng cao thì phải “hi sinh” tham số và bậc tự do trong tính toán.

Trong R, để tính SMA, chúng ta có thể dùng hàm **SMA()** trong package **TTR**. Hàm TTR đòi hỏi phải có tham số về order (hay span) với *n*. Chẳng hạn như nếu chúng ta muốn tính SMA 5 năm thì *n =* 5.

kings = scan("~/Documents/Data/kings.txt")

ts.kings = ts(kings)

library(TTR)

# tính trung bình cộng cho 5 năm

king.ma = SMA(ts.kings, n=5)

# vẽ biểu đồ trung bình cộng

plot.ts(king.ma)



**Biểu đồ tuổi thọ của vua Anh Quốc với SMA = 5**

Biểu đồ trên cho thấy vẫn còn khá nhiều độ dao động ngẫu nhiên với SMA=5. Để ước tính T chính xác hơn, chúng ta có thể tăng hệ số SMA lên 8:

king.ma8 = SMA(ts.kings, n=8)

plot.ts(king.ma8)



**Biểu đồ tuổi thọ của vua Anh Quốc với SMA = 8**

Biểu đồ trên cho thấy với SMA=8, chúng ta có một bức tranh rõ nét hơn về xu hướng. Chúng ta thấy rằng tuổi thọ của vua Anh có vẻ giảm từ 55 tuổi đến 38 tuổi trong 20 vị vua đầu, và tăng sau đó đến 73 tuổi (vị vua thứ 40).

### 22.3.2 Đối với dữ liệu có yếu tố seasonal

Như đề cập trên, các dữ liệu có yếu tố seasonal, chúng ta sẽ phân tích theo mô hình *Y*t = *S*t + *T*t + *E*t *.*  Chúng ta có thể dùng hàm decompose()có sẵn trong R cho phân tích. Quay lại ví dụ về số trẻ em mới sinh, chúng ta phân tích như sau:

births = scan(".../births.txt")

ts.births = ts(births, frequency=12, start=c(1946, 1))

comp = decompose(ts.births)

Hàm decompose()sẽ cung cấp cho chúng ta biết 3 thành phần trong dữ liệu như sau:

Phần liên quan đến seasonal (trích kết quả tiêu biểu):

$seasonal

Jan Feb Mar Apr May

1946 -0.6771947 **-2.0829607** 0.8625232 -0.8016787 0.2516514

1947 -0.6771947 -2.0829607 0.8625232 -0.8016787 0.2516514

Jun Jul Aug Sep Oct

1946 -0.1532556 **1.4560457** 1.1645938 0.6916162 0.7752444

1947 -0.1532556 1.4560457 1.1645938 0.6916162 0.7752444

Nov Dec

1946 -1.1097652 -0.3768197

1947 -1.1097652 -0.3768197

Chú ý trong phần seasonal, yếu tố seasonal cao nhất là tháng 7 (1.46) và thấp nhất là tháng 2 (-2.08). Các kết quả này cho thấy tháng có nhiều trẻ sinh nhất là tháng 7 và thấp nhất là tháng 2.

Phần liên quan đến trend cho từng năm và tháng:

$trend

Jan Feb Mar Apr May Jun Jul

1946 NA NA NA NA NA NA 23.98433

1947 22.35350 22.30871 22.30258 22.29479 22.29354 22.30562 22.33483

1948 22.43038 22.43667 22.38721 22.35242 22.32458 22.27458 22.23754

1949 22.06375 22.08033 22.13317 22.16604 22.17542 22.21342 22.27625

1950 23.21663 23.26967 23.33492 23.42679 23.50638 23.57017 23.63888

1951 24.00083 24.12350 24.20917 24.28208 24.35450 24.43242 24.49496

1952 24.27204 24.27300 24.28942 24.30129 24.31325 24.35175 24.40558

1953 24.78646 24.84992 24.92692 25.02362 25.16308 25.26963 25.30154

1954 25.92446 25.92317 25.92967 25.92137 25.89567 25.89458 25.92963

1955 25.64612 25.78679 25.93192 26.06388 26.16329 26.25388 26.35471

1956 27.21104 27.21900 27.20700 27.26925 27.35050 27.37983 27.39975

1957 27.44221 27.40283 27.44300 27.45717 27.44429 27.48975 27.54354

1958 27.68642 27.76067 27.75963 27.71037 27.65783 27.58125 27.49075

1959 26.96858 27.00512 27.09250 27.17263 27.26208 27.36033 NA

Aug Sep Oct Nov Dec

1946 23.66213 23.42333 23.16112 22.86425 22.54521

1947 22.31167 22.26279 22.25796 22.27767 22.35400

1948 22.21988 22.16983 22.07721 22.01396 22.02604

1949 22.35750 22.48862 22.70992 22.98563 23.16346

1950 23.75713 23.86354 23.89533 23.87342 23.88150

1951 24.48379 24.43879 24.36829 24.29192 24.27642

1952 24.44475 24.49325 24.58517 24.70429 24.76017

1953 25.34125 25.42779 25.57588 25.73904 25.87513

1954 25.98246 26.01054 25.88617 25.67087 25.57312

1955 26.40496 26.45379 26.64933 26.95183 27.14683

1956 27.44150 27.45229 27.43354 27.44488 27.46996

1957 27.56933 27.63167 27.67804 27.62579 27.61212

1958 27.46183 27.42262 27.34175 27.25129 27.08558

1959 NA NA NA NA NA

và sau cùng là phẫn ngẫu nhiên (chỉ liệt kê vài kết quả)

$random

Jan Feb Mar Apr May

1946 NA NA NA NA NA

1947 -0.237305288 0.863252404 0.543893429 0.175887019 -0.793193109

1948 0.183819712 -0.318705929 0.340268429 0.121262019 -0.354234776

1949 0.161444712 0.002627404 -0.571689904 -0.749362981 -0.666068109

1950 0.064569712 -0.292705929 0.479560096 1.047887019 1.561973558

Thay vì nhìn con số, chúng ta có thể vẽ biểu đồ các kết quả trên qua hàm plot như sau:

comp = decompose(ts.births)

plot(comp)



Biều đồ trên phân tích phần “observed” thành 3 phần: trend, seasonal, và random. Chúng ta thấy số trẻ em mới sinh giảm trong năm 1947 và sau đó thì tăng hàng năm, nhưng trong mỗi năm số trẻ em mới sinh biến đổi theo mùa. Còn phần ngẫu nhiên thì có vẻ độc lập với thời gian.

### Điều chỉnh cho ảnh hưởng của season

Một cách để “hiểu” xu hướng biến đổi là hiệu chỉnh cho yếu tố seasonal. Trong mô hình cộng hưởng, cách hiệu chỉnh đơn giản nhất là lấy hiệu số giữa giá trị quan sát và các tham số của yếu tố seasonal:

adj.births = ts.births-comp$seasonal

plot(adj.births)



Trong biểu đồ trên, phần ảnh hưởng của mùa đã được loại bỏ, và phần còn lại chỉ phản ảnh xu hướng – trend và phần ngẫu nhiên.

## 22.4 Tiên lượng bằng các phương pháp Exponential Smoothing

**22.4.1 Phương pháp simple exponential smoothing**

Nếu dữ liệu có thể mô tả bằng một mô hình cộng hưởng và *không có yếu tố mùa*, phương pháp *simple exponential smoothing* (SES, một số sách còn gọi là *single exponential smoothing*) là một cách tiên lượng trong một quãng thời gian ngắn. Để giải thích phương pháp này, tôi sẽ bắt đầu bằng những tiên lượng đơn giản.

Một cách “ngây thơ”, giá trị tiên lượng cho tương lai tuỳ thuộc vào giá trị quan sát trước đó. Gọi giá trị tương lai là *Y*t+1 và giá trị hiện tại là *Y*t, phát biểu trên có nghĩa là:

*Y*t+1 = *Y*t

Một cách ngây thơ khác là dùng trung bình trong quá khứ tính đến thời điểm *t*.

Nhưng phương pháp trung bình này giả định rằng tất cả các giá trị quá khứ có trọng số như nhau. Giả định này không hợp lí, vì đối với tiên đoán, các giá trị gần nhất phải có trọng số cao hơn các giá trị cũ. Do đó, phương pháp SES phát biểu rằng:

Trong đó, α được gọi là *smoothing parameter.* Do đó, mượt mà hoá (smoothing) được kiểm soát bằng một trọng số α (cũng có thể xem là thông số) có giá trị từ 0 đến 1. Alpha bằng 0 hay gần 0 có nghĩa là trọng số cho số liệu trước đó không cao trong tiên lượng giá trị kế tiếp. Để thấy ý nghĩa đó, chúng ta thử xem qua vài giá trị trong bảng sau:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Giá trị** | **Αlpha= 0.2** | **Αlpha= 0.4** | **Αlpha= 0.6** | **Αlpha= 0.8** |
| *Y*t | 0.2 | 0.4 | 0.6 | 0.8 |
| *Y*t-1 | 0.16 | 0.24 | 0.24 | 0.16 |
| *Y*t-2 | 0.128 | 0.144 | 0.096 | 0.032 |
| *Y*t-3 | 0.1024 | 0.0864 | 0.0384 | 0.0064 |
| *Y*t-4 | (0.2)(0.8)4 | (0.4)(0.6)4 | (0.6)(0.4)4 | (0.8)(0.2)4 |
| *Y*t-5 | (0.2)(0.8)5 | (0.4)(0.6)5 | (0.6)(0.4)5 | (0.8)(0.2)5 |

Để minh hoạ cho tiên lượng bằng phương SES, tôi sẽ dùng dữ liệu về lượng mưa hàng năm từ 1813 đến 1912 (của Hipel và McLeod):

# Đọc dữ liệu, bỏ dòng đầu

rain = scan("~/Documents/Data/rainfalls.txt", skip=1)

ts.rain = ts(rain, start=c(1813))

plot.ts(ts.rain)

Để tiên lượng bằng phương pháp, chúng ta dùng hàm HoltWinters() với tham số beta=FALSE và gamma=FALSE (tham số beta và gamma parameters được dùng cho mục đích khác).

forecast = HoltWinters(ts.rain, beta=F, gamma=F)

forecast

Hàm HoldWinters cho kết quả alpha = 0.024 và Coefficients = 24.678. Vì alpha gần bằng 0, các giá trị tiên lượng dựa vào các giá trị gần nhất không có trọng số cao. Hàm HoldWinters cho ra kết quả tiên lượng trong một biến / đối tượng có tên là **fitted**. Chúng ta có thể xem qua giá trị tiên lượng bằng bảng số liệu và biểu đồ:

forecast$fitted

plot(ts.rain); plot(forecast)

sẽ cho ra kết quả (trích):

> forecast$fitted

Time Series:

Start = 1814

End = 1912

Frequency = 1

xhat level

1814 23.56000 23.56000

1815 23.62054 23.62054

1816 23.57808 23.57808

1817 23.76290 23.76290

…

1910 24.57541 24.57541

1911 24.59433 24.59433

1912 24.59905 24.59905



Biểu đồ cho thấy giá trị gốc và tiên lượng (đường màu đỏ) của lượng mưa ở London từ năm 1814 đến 1912. Cũng như mô hình hồi qui tuyến tính, chúng ta cũng có thể tính toán các chỉ số phản ảnh độ chính xác như tổng bình phương sai số (sum of squared errors) bằng cách trích SSE từ đối tượng forecast:

forecast$SSE

Như đề cập trên, hàm HoltWinters() chỉ tiên lượng trong khoản thời gian của dữ liệu gốc. Nhưng trong thực tế, chúng ta muốn tiên lượng trong tương lai. Để tiên lượng giá trị tương lai chúng ta có thể dùng hàm forecast.HoltWinters() trong package chuyên dụng forecast. Chẳng hạn như chúng ta muốn tiên lượng lượng mưa trong 5 năm tới, chúng ta dùng hàm sau đây:

library(forecast)

forecast2 = forecast.HoltWinters(forecast, h=5)

> forecast2

Point Forecast Lo 80 Hi 80 Lo 95 Hi 95

1913 24.67819 19.17493 30.18145 16.26169 33.09470

1914 24.67819 19.17333 30.18305 16.25924 33.09715

1915 24.67819 19.17173 30.18465 16.25679 33.09960

1916 24.67819 19.17013 30.18625 16.25434 33.10204

1917 24.67819 19.16853 30.18785 16.25190 33.10449

Hàm HoltWinters cung cấp giá trị tiên lượng và khoảng tin cậy 80%, 95%. Chẳng hạn như lượng mưa cho năm 1917 được dự đoán là 24.68 inches, với khoảng tin cậy 95% từ 16.25 đến 33.10. Một cách dễ nhìn hơn là dùng biểu đồ:

plot.forecast(forecast2)



Biểu đồ trên vẽ các giá trị tiên lượng bằng màu xanh, với khoảng tin cậy 80% và 95% trong vùng màu xám.

### 22.4.2 Tiên lượng bằng phương pháp Holt Exponential Smoothing

Một phương pháp khác có ứng dụng cho các dữ liệu không có yếu tố mùa là phương pháp Holt Exponential Smoothing (HES). Phương pháp HES ứng dụng cho *tiên lượng ngắn hạn* khi dữ liệu có xu hướng tăng hoặc giảm (nhưng không có yếu tố mùa).

Phương pháp HES được hình thành từ 2 thông số smoothing là alpha và beta, có giá trị từ 0 đến 1. Alpha là trọng số của giá trị tại thời điểm hiện tại. Beta là độ dốc (slope) cho phần trend ngay tại thời điểm hiện tại. Một ví dụ ứng dụng của phương pháp HES là dữ liệu về đường kính của đường viền váy phụ nữ đo lường từ 1866 đến 1911.

skirts = scan("~/Documents/Data/skirts.txt", skip=5)

ts.skirts = ts(skirts, start=c(1866))

plot.ts(ts.skirts)



Chúng ta thấy rằng đường kính váy tăng từ 1866 đến 1880 (tăng khoảng 600), và sau đó thì giảm đến 1911 (giảm ~520). Để tiên lượng, chúng ta cần phải dùng hàm

HoltWinters() để phân tích dữ liệu.

forecast1 = HoltWinters(ts.skirts, gamma=F)

forecast1

Cho ra thông số alpha = 0.84 và beta = 1. Hai thông số này cho chúng ta biết giá trị hiện hành có trọng số khá cao hay rất cao trong việc tiên lượng giá trị tương lai. Điều này khá nhất quán với biểu đồ dưới đây phản ảnh giá trị quan sát và giá trị tiên lượng rất nhất quán nhau:

plot(forecast1)



Cũng như phương pháp SMS, với phương pháp HES, chúng ta cũng có thể tiên lượng cho một tương lai ngắn hạn bằng hàm forecast.HoltWinters() package chuyên dụng forecast. Chẳng hạn như tiên đoán cho 10 năm:

forecast1 = HoltWinters(ts.skirts, gamma=F)

forecast2 = forecast.HoltWinters(forecast1, h=10)

plot.forecast(forecast2)



Trong biểu đồ trên, đường màu xanh thể hiện giá trị tiên lượng, cùng khoảng tin cậy 95% và 80% (vùng màu xanh và màu xám).

### 22.4.3 Tiên lượng bằng phương pháp Holt-Winters Exponential Smoothing

Nếu dữ liệu có thể mô tả bằng một mô hình cộng hưởng, nhưng **có yếu tố** mùa, phương pháp Holt-Winters Exponential Smoothing có thể dùng cho tiên lượng ngắ hạn. Phương pháp này dùng 3 thông số smoothing (α, β và γ) để thể hiện dữ liệu. Thông số alpha là trọng số của giá trị hiện tại; thông số beta thể hiện độ dốc của giá trị hiện tại; và thông số gamma thể hiện mùa - seasonality. Dĩ nhiên, tất cả các thông số α, β và γ đều có giá trị giữa 0 và 1.

Để minh hoạ cho phương pháp Hold-Winter Exponential Smoothing, chúng ta quay lại với ví dụ về lượng hàng lưu niệm bán ra từ một cửa hàng ở Queensland. Chú ý chúng ta dùng đơn vị log.

souv = scan("~/Documents/Data/souvenirs.txt")

ts.souv = ts(souv, frequency=12, start=c(1987, 1))

log.souv = log(ts.souv)

forecast1 = HoltWinters(log.souv)

Chúng ta có hệ số α = 0.413, β = 0, và Υ = 0.956. Giá trị của thông số alpha (0.41) tương đối thấp, cho thấy giá trị hiện tại phụ thuộc vào các giá trị gần đây và một số giá trị trong quá khứ khá xa. Thông số beta = 0 cho thấy độ dốc của trend không có “cập nhật hoá” trong thời gian qua. Thông số gamma = 0.956 rất cao, cho thấy ảnh hưởng của yếu tố mùa tại thời điểm hiện tại phụ thuộc vào các giá trị gần nhất. Chúng ta có thể vẽ biểu đồ để biết giá trị tiên lượng và thực tế rất gần nhau, chứng tỏ mô hình tiên lượng rất tốt.

plot(forecast1)



Chúng ta cũng có thể tiên lượng cho tương lai 5 năm (tức 60 tháng):

forecast1 = HoltWinters(log.souv)

forecast2 = forecast.HoltWinters(forecast1, h=60)

plot.forecast(forecast2)



Như thường lệ, hàm plot.forecast vẽ giá trị tiên lượng bằng màu xanh, kèm theo khoảng tin cậy 80% và 95% trong vùng màu xám nhạt và xám đậm.

## 22.5 Mô hình ARIMA

Thật ra, phải nói là “các mô hình ARIMA” mới đúng, vì đó là tập hợp một số mô hình phân tích và tiên lượng cho dữ liệu theo thời gian. Các phương pháp tôi vừa mô tả trên đây (như Simple Exponential Smoothing, Holt Exponential Smoothing, Holt-Winters Exponential Smoothing) là những *phương pháp tiên lượng* mà không dựa vào những giả định về mối tương quan giữa các giá trị liên tiếp nhau trong dãy số liệu (còn gọi là ***autocorrelation***, có lẽ tạm dịch là *tương quan liên hoàn*). Tuy nhiên, nếu chúng ta muốn tiên lượng khoảng tin cậy bằng các phương pháp exponential smoothing thì giả định là sai số (error) của các giá trị tiên lượng không liên quan nhau và tuân theo luật phân bố chuẩn với trung bình 0 và phương sai bất biến.

Mặc dù các phương pháp exponential smoothing không đặt giả định về tương quan liên hoàn, trong một số trường hợp, giá trị tiên lượng sẽ chính xác hơn nếu hiệu chỉnh cho hệ số tương quan. Các mô hình “Autoregressive Integrated Moving Average” hay ARIMA là những mô hình thống kê dùng để **mô hình hoá phần ngẫu nhiên** của một dãy số liệu theo thời gian. Trong các mô hình ARIMA, hệ số tương quan liên hoàn được điều chỉnh.

### 22.5.1 Khái niệm *sai phân* (differencing)

**Dữ liệu ổn định và dữ liệu không ổn định**

Các mô hình ARIMA được ứng dụng thường xuyên cho các dãy dữ liệu theo thời gian ổn định (thuật ngữ tiếng Anh là *stationary time series*). Trong thống kê, dữ liệu theo thời gian ổn định (sẽ gọi tắt là **dữ liệu ổn định**) có nghĩa là dữ liệu mà các chỉ số thống kê như trung bình, phương sai, hệ số tương quan liên hoàn, v.v. đều không thay đổi theo thời gian.

Ngược lại, khi các chỉ số thống kê, nhất là trung bình và phương sai, có xu hướng biến chuyển theo thời gian, thì chúng ta có một **dữ liệu bất ổn định** -- *non-stationary time series*. Với dữ liệu bất ổn định, chúng ta phải bắt đầu bằng cách tính *sai phân* (difference) dữ liệu cho đến khi nó trở thành ổn định.

**Sai phân bậc 1** của một dãy số liệu theo thời gian là hiệu số của số liệu tại thời điểm *t* và số liệu của một thời đểm *t* - 1. Nếu *Y*t là giá trị tại thời điểm *t*, và *Y*t-1 tại thời điểm *t*-1, thì sai phân tại thời điểm *t* chính là *Y*t – *Y*t-1. Sai phân này có khi còn gọi là *first difference*.

Nếu sai phân bậc 1 của *Y* ổn định dữ liệu và hoàn toàn ngẫu nhiên (tức không có autocorrelation), thì *Y* còn được mô tả bằng mô hình *bước đi ngẫu nhiên* – tức *random walk model*.

Nếu sai phân bậc 1 của *Y* ổn định nhưng không ngẫu nhiên (tức là giá trị tại thời điểm *t* có liên quan với giá trị trước) thì cần phải có một **mô hình** **ARIMA** để mô tả dữ liệu.

Nếu chúng ta phải tính sai phân dữ liệu *d* lần (hay bậc sai phân bậc *d*) để có được dữ liệu ổn định, thì chúng ta có mô hình ARIMA(p, d, q); trong đó, *d* là bậc thứ tự [order] của sai phân. Trong R, có hàm **diff()** với tham số **differences = d** có thể dùng để tính sai phân. Chẳng hạn như đối với dữ liệu về đường kính của váy từ 1866 đến 1911 là dữ liệu không ổn định vì trung bình biến chuyển theo thời gian (xem biểu đồ dưới đây):



Chúng ta có thể tính sai phân cho dữ liệu trên:

skirts = scan("~/Documents/Data/skirts.txt", skip=5)

ts.skirts = ts(skirts, start=c(1866))

# tính sai phân

dif.skirts = diff(ts.skirts, differences=1)

plot.ts(dif.skirts)



Kết quả cho thấy dữ liệu có vẻ không ổn định vì trung bình thay đổi theo thời gian. Chúng ta thử sai phân bậc 2:

dif.skirts2 = diff(ts.skirts, differences=2)

plot.ts(dif.skirts2)



Biểu đồ trên cho thấy sai phân bậc 2 đã làm cho dữ liệu ổn định (không có xu hướng tăng/giảm theo thời gian). Phương sai cũng có vẻ ổn định. Như vậy, sai phân bậc 2 (*d =* 2) có vẻ thích hợp cho dữ liệu.

Do đó, mô hình ARIMA bây giờ là ARIMA(p, 2, q). Vấn đề kế tiếp là xác định tham số *p* và *q*. Quay lại ví dụ về tuổi thọ của các vị vua Anh Quốc, chúng ta thấy dữ liệu không phải thuộc loại ổn định.



Bây giờ, chúng ta tính sai phân bậc 1 cho dữ liệu:

kings = scan("~/Documents/Data/kings.txt")

ts.kings = ts(kings)

dif.kings = diff(ts.kings, difference=1)

plot.ts(dif.kings)



Biểu đồ trên cho thấy dữ liệu đã ổn định với sai phân bậc 1. Mô hình ARIMA(p, 1, q) có lẽ thích hợp cho dữ liệu về tuổi thọ của các vị vua Anh Quốc.

Hai ví dụ trên cho thấy khi tính sai phân, chúng ta loại bỏ phân trend của dữ liệu theo thời gian. Phần còn lại chính là phần ngẫu nhiên hay irregular hay random. Bây giờ chúng ta sẽ xem xét đến hệ số tương quan liên hoàn xem có thể giúp gì cho chúng ta mô hình hoá dữ liệu.

### 22.5.2 Chọn mô hình ARIMA

Mô hình ARIMA(p, d, q) có ít nhất 3 thông số. Chúng ta đã thấy cách chọn / xác định thông số *d.* Nhưng thông số *p* và *q* có thể xác định qua phân tích tương biểu đồ quan liên hoàn (correlogram) và biểu đồ tương quan liên hoàn từng phần (partial correlogram) của dữ liệu ổn định.

Khái niệm **autocorrelation – tương quan liên hoàn** có thể giải thích ngắn gọn như sau. Gọi *Y*1, *Y*2, *Y*3, …, *Y*N là một dãy số liệu theo thời gian được ghi nhận ở thời điểm *t*1, *t*2, *t*3, …, *t*N. Autocorrelation được định nghĩa là:

Như vậy, autocorrelation chính là một hệ số tương quan. Nhưng thay vì tương quan giữa 2 biến X và Y, thì autocorrelation là tương quan giữa *Yi* và *Yi+k*. Autocorrelation bậc 1 (tức *k*=1) thường được sử dụng để xác định hiện tượng phi ngẫu nhiên (non-randomness). Trong R có hàm **acf()** và **pccf()** có thể dùng để tính toán hệ số autocorrelation và hệ số autocorrelation từng phần.

Chúng ta dùng dữ liệu về tuổi thọ của các vua Anh Quốc làm ví dụ ước tính autocorrelation và autocorrelation từng phần, và qua đó xác định thông số p và q cho mô hình ARIMA.

kings = scan("~/Documents/Data/kings.txt")

ts.kings = ts(kings)

dif.kings = diff(ts.kings, difference=1)

acf(dif.kings, lag.max=20, plot=F)

acf(dif.kings, lag.max=20)

Chúng ta sẽ có kết quả như sau:

> acf(dif.kings, lag.max=20, plot=F)

Autocorrelations of series ‘dif.kings’, by lag

0 1 2 3 4 5 6 7 8 9

1.000 -0.360 -0.162 -0.050 0.227 -0.042 -0.181 0.095 0.064 -0.116

10 11 12 13 14 15 16 17 18 19

-0.071 0.206 -0.017 -0.212 0.130 0.114 -0.009 -0.192 0.072 0.113

20

-0.093

hay thể hiện qua biểu đồ correlogram:



Chúng ta thấy hệ số autocorrelation ở sai phân bậc 1 (-0.360) quá ngưỡng ý nghĩa thống kê. Ngưỡng ý nghĩa thống kê là hai đường đứt đoạn màu xanh trong biểu đồ. Tuy nhiên, hệ số tương quan của các bậc khác thì đều 0, vì nằm trong vùng không có ý nghĩa thống kê.

Tương tự, chúng ta xem xét hệ số autocorrelation từng phần bằng hàm pacf():

pacf(dif.kings, lag.max=20, plot=F)

pacf(dif.kings, lag.max=20)

> pacf(dif.kings, lag.max=20, plot=F)

Partial autocorrelations of series ‘dif.kings’, by lag

1 2 3 4 5 6 7 8 9 10

-0.360 -0.335 -0.321 0.005 0.025 -0.144 -0.022 -0.007 -0.143 -0.167

11 12 13 14 15 16 17 18 19 20

0.065 0.034 -0.161 0.036 0.066 0.081 -0.005 -0.027 -0.006 -0.037



Biểu đồ và kết quả trên cho thấy hệ số autocorrelation từng phần ở bậc (lags hay sai phân) 1, 2 và 3 vượt qua ngưỡng ý nghĩa thống kê và âm (lag 1: -0.360, lag 2: -0.335, lag 3:-0.321). Tuy nhiên, hệ số này giảm dần và gần như 0 sau lag 3.

Qua phân tích autocorrelation và autocorrelation từng phần như trên, chúng ta có thể xác định 2 thông số p và q cho mô hình ARMA (autoregressive moving average) cho dữ liệu theo thời gian với sai phân bậc 1:

* Mô hình ARMA(3,0), tức là mô hình autoregressive với p=3 có vẻ là tối ưu, vì autocorrelogram từng phần bằng 0 sau lag 3.
* Mô hình ARMA(0,1), tức là mô hình moving average với q=1, vì autocorrelogram bằng zero sau lag 1.

Ở đây, chúng ta cần dùng *nguyên lí tiết kiệm* (Principle of Parsimony) để quyết định mô hình nào là tốt nhất. Nói cách khác, chúng ta muốn tìm mô hình có ít thông số nhất nhưng mô tả dữ liệu tốt nhất. Mô hình ARMA(3, 0) có 3 thông số, mô hình ARMA(0, 1) có 1 thông số, và mô hình chung ARMA(p, q) có ít nhất là 2 thông số. Do đó, mô hình ARMA(0,1) trong trường hợp này là mô hình tốt nhất.

**Mô hình Moving Average (MA model)**

Mô hình MA chủ yếu quan tâm đến white noise (*Zt*) chứ không phải *Y*t*.* Mô hình MA bậc *q,* viết tắt là MA(q), có thể phát biểu như sau:

*Y*t = *Z*t + θ1*Z*t-1 + θ2*Z*t-2 + θ3*Z*t-3 + ... + θq*Z*t-q

Trong đó, θ1, θ2, θ3, ..., θq là thông số của mô hình. Mô hình này có *q* sai phân(lag), và *Z*t là white noise.

Mô hình ARMA(0, 1) thực chất là một mô hình moving average – trung bình động bậc 1. Nói cách khác, đó là **mô hình MA(1).** Mô hình này có thể viết như sau:

*Y*t – μ = *Z*t – (θ×*Z*t-1)

trong đó, *Y*t là một dãy số liệu ổn định và chúng ta nghiên cứu sai phân bậc 1; μ là trung bình *Y*t, *Z*t là yếu tố ngẫu nhiên với trung bình 0 và phương sai bất biến (*Z*t còn gọi là “white noise”); và θ là thông số cần ước tính.

Mô hình MA (moving average) thường được sử dụng để mô hình dữ liệu theo thời gian với đặc điểm là các số liệu phụ thuộc nhau trong một khoản thời gian ngắn. Trong thực tế, mô hình MA có thể dùng để mô ta phần ngẫu nhiên của tuổi thọ của các vua, vì chúng ta kì vọng rằng tuổi thọ của một vị vua có thể có ảnh hưởng đến tuổi thọ của vị vua kế tiếp hay vài vị vua tiếp theo, nhưng khó có thể ảnh hưởng đến các vua sau một thời gian dài.

**Tìm thông số cho mô hình ARIMA bằng hàm auto.arima()**

Các phân tích trên đây để tìm thông số cho mô hình ARIMA(p, d, q) có thể nói là mang tính thủ công. Trong thực tế, chúng ta có thể sử dụng hàm auto.arima() trong package chuyên dụng **forecast** để tìm các thông số từ dữ liệu gốc. Để “kiểm tra” hàm auto.arima(), chúng ta thử mô phỏng một dãy dữ liệu theo thời gian với mô hình ARIMA(1, 0, 1), với thông số p = 0.3 và q = 0.6. Mô phỏng có thể dùng hàm arima.sim có sẵn trong R:

test.data = arima.sim(model=list(ar=0.3, ma=0.6), n=1000, n.start=200)

Hàm trên mô phỏng 1000 giá trị theo mô hình AR(0.3) và MA(0.6). Lệnh n.stat=200 có nghĩa là giá trị mô phỏng chí lấy từ 201 trở lên để đảm bảo ổn định của mô phỏng.

Bây giờ chúng ta thử dùng hàm auto.arima() để tìm mô hình tốt nhất:

library(forecast)

auto.arima(test.data)

Kết quả cho thấy:

**> auto.arima(test.data)**

Series: test.data

ARIMA(1,0,1) with zero mean

Coefficients:

ar1 ma1

0.3641 0.5797

s.e. 0.0377 0.0330

Kết quả đề nghị mô hình ARIMA(1, 0, 1), tức là đúng với mô hình chúng ta mô phỏng. Thông số là AR(0.36) gần bằng với giá trị thật 0.30, và MA(0.58) gần với giá trị thật là 0.60. Dĩ nhiên giá trị mô phỏng không thể nào chính xác với giá trị thật, nhưng kết quả của hàm auto.arima() rõ ràng là đề nghị mô hình hợp lí.

Bây giờ, chúng ta quay lại với dữ liệu thực tế để dùng auto.arima() tìm thông số cho dữ liệu về tuổi thọ của vua:

kings = scan("~/Documents/Data/kings.txt")

library(forecast)

auto.arima(kings)

Với kết quả:

> auto.arima(kings)

Series: kings

ARIMA(0,1,1) with drift

Coefficients:

ma1 drift

-0.7463 0.3882

s.e. 0.1294 0.6603

Kết qua trên cho thấy mô hình thích hợp là ARIMA(0, 1, 1). Nói cách khác, thông số của mô hình là p=0, d=1, q=1. Mô hình này nhất quán với phân tích thủ công mà chúng ta đã thực hiện.

**Mô hình autoregression (AR model)**

Ý tưởng của mô hình autoregressive (AR) là giá trị quan sát hiện tại phụ thuộc vào các giá trị trước đó. Gọi *Zt* là một white noise. Nếu chúng ta mô hình *Zt* bằng một phương trình bậc 2 như sau:

*Y*t = *Y*t-1 – 0.9*Y*t-2 + *Z*t

Phương trình này phát biểu rằng giá trị của thời điểm hiện tại phụ thuộc vào (a) giá trị của thời điểm trước hiện nay, và (b) giá trị của thời điểm trước đó. Đây là mô hình autoregressive (AR).

**Ví dụ**: **Khẩu trang che mặt chống bụi núi lửa ở Bắc bán cầu**. Dữ liệu sau đây bao gồm chỉ số về khẩu trang chống bụi được đo lường từ 1500 đến 1969 tại vùng Bắc bán cầu (dữ liệu của Hipel và Mcleod, 1994). Chúng ta đọc dữ liệu vào R và hoán chuyển sang dữ liệu theo thời gian với hàm ts như sau:

vc = scan("~/Documents/Data/vulcano.txt", skip=1)

ts.vc = ts(vc, start=c(1500))

plot.ts(ts.vc)



Biểu đồ trên cho thấy những dao động ngẫu nhiên của số liệu có vẻ bất biến theo thời gian. Do đó, một mô hình cộng hưởng có vẻ thích hợp cho việc mô tả dữ liệu. Ngoài ra, biểu đồ trên cũng cho thấy phương sai độc lập với thời gian. Do đó, chúng ta không cần phải tính sai phân cho dữ liệu để dùng mô hình ARIMA, nhưng có thể áp dụng mô hình ARIMA trực tiếp vào dữ liệu gốc (tức *d* = 0). Chúng ta tìm các thông số bằng hàm auto.arima() như sau:

> auto.arima(vc)

Series: vc

ARIMA(1,0,2) with non-zero mean

Coefficients:

ar1 ma1 ma2 intercept

0.4723 0.2694 0.1279 57.5178

s.e. 0.0936 0.0969 0.0752 8.4883

Kết quả trên “đề nghị” mô hình ARIMA(1, 0, 2). Mô hình có 3 thông số. Tuy nhiên, nếu chúng ta dùng tiêu chuẩn BIC (Bayesian Information Criterion) thì mô hình có lẽ đơn giản hơn:

> **auto.arima(vc, ic="bic")**

Series: vc

ARIMA(2,0,0) with non-zero mean

Coefficients:

ar1 ar2 intercept

0.7533 -0.1268 57.5274

s.e. 0.0457 0.0458 8.5958

Với tiêu chuẩn BIC, chúng ta có mô hình ARMA(2, 0). Mô hình này có 2 thông số, và so với các mô hình khác, đây là mô hình tối ưu. Đây là **mô hình autoregressive** bậc 2, hay AR(2). Mô hình này phát biểu rằng

*Y*t – μ = β1(*Y*t-1 – μ) + β2(*Y*t-2 – μ) + *Z*t

Trong đó *Y*t là số liệu tại thời điểm *t,* μ là số trung bình của dãy số liệu, β1 và β2 là thông số cần tìm, và *Z*t là yếu tố ngẫu nhiên (white noise) với trung bình 0 và phương sai bất biến.

**Model AR (autoregressive)** thường được sử dụng để mô tả dãy số liệu theo thời gian mà sự tương quan giữa các số liệu nối tiếp nhau *mang tính lâu dài*. Mô hình này thích hợp cho những trường hợp như số liệu về bụi núi lửa vì chúng ta kì vọng rằng bụi núi lửa của một năm còn ảnh hưởng đến nhiều năm sau, bởi vì bụi không thể nào biến mất một cách nhanh chóng.

Nếu mô hình ARMA(2, 0), tức p=2, q=0 được dùng để mô tả dữ liệu bụi núi lửa, thì điều này có nghĩa là chúng ta có mô hình ARIMA(2, 0, 0), tức p=2, d=0, và q=0.

### 22.5.3 Tiên lượng dùng mô hình ARIMA

Sau khi xác định mô hình tốt nhất, bước kế tiếp là ước tính thông số và dùng thông số để làm một mô hình tiên lượng các số liệu tương lai. Trong R có hàm arima() dùng để ước tính thông số của mô hình ARIMA.

Chúng ta dùng số liệu về tuổi lúc tử vong của các vua Anh Quốc làm ví dụ cho mô hình ARIMA. Nhớ rằng chúng ta đã có mô hình “tối ưu” cho dữ liệu là ARIMA(0, 1, 1). Trong R, chúng ta dùng lệnh order=c(p, d, q):

kings = scan("~/Documents/Data/kings.txt")

ts.kings = ts(kings)

arima1 = arima(ts.kings, order=c(0, 1, 1))

Với kết quả như sau:

> arima1

Series: ts.kings

ARIMA(0,1,1)

Coefficients:

ma1

-0.7218

s.e. 0.1208

sigma^2 estimated as 230.4: log likelihood=-170.06

AIC=344.13 AICc=344.44 BIC=347.56

Như đề cập trên, nếu chúng ta áp dụng mô hình ARIMA(0, 1, 1), thì điều đó cũng tương đương với áp dụng mô hình ARMA(0, 1) vào dữ liệu sai phân bậc 1. Mô hình ARMA(0, 1) có thể viết như sau:

*Y*t – μ = *Z*t –θ*Z*t-1

Trong kết quả phân tích trên, θ = -0.7218.

Chúng ta cũng có thể tiên lượng bằng cách dùng hàm forecast.Arima() trong package forecast:

library(forecast)

forecast1 = forecast.Arima(arima1, h=5)

forecast1

plot.forecast(forecast1)

Kết quả trình bày cho 5 năm tới với khoảng tin cậy 80% và 95%.

> forecast1

Point Forecast Lo 80 Hi 80 Lo 95 Hi 95

43 67.75063 48.29647 87.20479 37.99806 97.50319

44 67.75063 47.55748 87.94377 36.86788 98.63338

45 67.75063 46.84460 88.65665 35.77762 99.72363

46 67.75063 46.15524 89.34601 34.72333 100.77792

47 67.75063 45.48722 90.01404 33.70168 101.79958



Tuổi thọ của vua thứ 42 là 56 tuổi. Mô hình ARIMA tiên lượng rằng vị vua 43 sẽ thọ 67.8 tuổi.

**Tiên lượng dữ liệu bụi núi lửa**. Quay trở lại ví dụ về bụi núi lửa, chúng ta đã xác định mô hình tối ưu là ARIMA(2, 0, 0). Để ước tính thông số mô hình này, chúng ta dùng hàm arima:

vc = scan("~/Documents/Data/vulcano.txt", skip=1)

ts.vc = ts(vc, start=c(1500))

arima1 = arima(ts.vc, order=c(2,0,0))

arima1

Với kết quả:

> arima1

Series: ts.vc

ARIMA(2,0,0) with non-zero mean

Coefficients:

ar1 ar2 intercept

0.7533 -0.1268 57.5274

s.e. 0.0457 0.0458 8.5958

sigma^2 estimated as 4870: log likelihood=-2662.54

AIC=5333.09 AICc=5333.17 BIC=5349.7

Như đề cập trên, mô hình ARIMA(2,0,0) cũng có thể viết là:

*Y*t – μ = β1(*Y*t-1 – μ) + β2(*Y*t-2 – μ) + *Z*t

Kết quả trên cho thấy ước số của β1 = 0.753 và β2 = -0.127. Bây giờ, chúng ta có thể tiên lượng cho 31 năm sau bằng hàm forecast.Arima():

forecast1 = forecast.Arima(arima1, h=31)

forecast1

plot.forecast(forecast1)

> forecast1

Point Forecast Lo 80 Hi 80 Lo 95 Hi 95

1970 21.48131 -67.94860 110.9112 -115.2899 158.2526

1971 37.66419 -74.30305 149.6314 -133.5749 208.9033

1972 47.13261 -71.57070 165.8359 -134.4084 228.6737

1973 52.21432 -68.35951 172.7881 -132.1874 236.6161

1974 54.84241 -66.22681 175.9116 -130.3170 240.0018

1975 56.17814 -65.01872 177.3750 -129.1765 241.5327

1976 56.85128 -64.37798 178.0805 -128.5529 242.2554

1977 57.18907 -64.04834 178.4265 -128.2276 242.6057

1978 57.35822 -63.88124 178.5977 -128.0615 242.7780

1979 57.44283 -63.79714 178.6828 -127.9777 242.8634

...

1998 57.52739 -63.71275 178.7675 -127.8934 242.9482

1999 57.52739 -63.71275 178.7675 -127.8934 242.9482

2000 57.52739 -63.71275 178.7675 -127.8934 242.9482

Tóm lại, qua chương này, các bạn đã làm quen với một số khái niệm và phương pháp cơ bản thường hay dùng trong phân tích dữ liệu theo thời gian. Một số phương pháp tuy nói là “cơ bản” nhưng khá phức tạp nếu mô tả bằng ngôn ngữ toán. Do đó, tôi cố gắng không trình bày các công thức toán, mà thay vào đó là ý tưởng và cách dùng R để ước tính các thông số của mô hình AR, MA, và ARIMA. Tôi hi vọng rằng sau khi đọc và hiểu hết chương này, các bạn có thể thực hiện những phân tích dữ liệu theo thời gian và có thể diễn giải kết quả một cách thuyết phục.

**Chú thích**

Sách về phân tích dữ liệu theo thời gian dùng R rất phong phú. Chương này được mô phỏng theo và dùng dữ liệu từ cuốn “*Forecasting: principles and practice*” của R. Hyndman và [G. Athanasopoulos](http://www.amazon.com/s/ref=dp_byline_sr_book_2?ie=UTF8&field-author=George+Athanasopoulos&search-alias=books&text=George+Athanasopoulos&sort=relevancerank) (Otexts, 2012; Amazon.com). Ngoài ra, cuốn “*Times Series Analysis and Its Applications (with R examples)*” của Robert Shumway và David Stoffer (Nhà xuất bản Springer) cũng rất có ích và dễ hiểu.