基于Python的时间序列分析

白晓东　编著

清華大學出版社
北京

内 容 简 介

本书在借鉴国内外相关教材优点的基础上，总结作者多年讲授时间序列分析课程的教学经验和体会，本着“教师好用、学生好读”的指导思想，系统地介绍了一元时间序列分析的基本思想、基本原理和基本方法，内容包括时间序列的基本概念、时间序列数据的预处理方式、分解和平滑、趋势的消除、单位根检验和协整、平稳时间序列模型、非平稳时间序列模型、残差自回归模型、季节模型、异方差时间序列模型、谱分析、基于深度学习的时间序列预测以及上述模型的性质、建模、预测，此外还包含了大量的实例. 本书全程使用 Python 语言分析了来自不同学科的真实数据.

本书通俗易懂，理论与应用并重，可作为高等院校统计、经济、商科、工程以及定量社会科学等相关专业的高年级本科生学习时间序列分析的教材或教学参考书，也可作为硕士研究生使用 Python 语言学习时间序列分析的入门书，还可供相关技术人员进行时间序列数据处理时参考书.

图书在版编目(CIP)数据

基于 Python 的时间序列分析 / 白晓东编著.—北京：清华大学出版社，2023.4
ISBN 978-7-302-62684-8

Ⅰ.①基… Ⅱ.①白… Ⅲ.①软件工具—程序设计—应用—时间序列分析—高等学校—教材
Ⅳ.①O211.61-39

中国国家版本馆 CIP 数据核字(2023)第 025958 号

责任编辑：刘 颖
封面设计：傅瑞学
责任校对：王淑云
责任印制：沈 露

出版发行：清华大学出版社
网 址：http://www.tup.com.cn, http://www.wqbook.com
地 址：北京清华大学学研大厦 A 座　　邮 编：100084
社 总 机：010-83470000　　邮 购：010-62786544
投稿与读者服务：010-62776969, c-service@tup.tsinghua.edu.cn
质量反馈：010-62772015, zhiliang@tup.tsinghua.edu.cn
印 装 者：三河市龙大印装有限公司
经 销：全国新华书店
开 本：185mm × 260mm　　**印 张**：18.25　　**字 数**：439 千字
版 次：2023 年 4 月第 1 版　　**印 次**：2023 年 4 月第 1 次印刷
定 价：58.00 元

产品编号：093904-01

前　言

时间序列分析是一种处理动态数据的统计方法. 它是基于随机过程理论和数理统计方法而发展起来的, 是寻找动态数据的变化特征, 挖掘隐含信息, 建立拟合模型, 进而预测数据未来发展的有力统计工具, 它广泛应用于经济、金融、气象、天文、物理、化学、生物、医学、质量控制等社会科学、自然科学和生产实践的诸多领域, 已经成为许多行业常用的统计方法.

目前, 国内外有关时间序列分析的教材已有很多, 其中一些偏重于理论的讲述, 需要读者具备比较深厚的概率论与数理统计基础, 主要阅读对象是精英型的统计学专业的学生; 另一些则侧重于模型的应用, 不关注理论和技术细节的推导, 主要阅读对象是经管类专业的学生. 随着我国招生制度的变化和大数据产业的飞速发展, 大部分高校的统计学及其相关专业的培养目标逐步转为复合应用型人才的培养, 强调培养具有数据分析能力的人才的重要性. 显然那些过于偏重理论讲述和过于偏重模型应用的教材不能适应这一变化.

为适应培养要求的转变, 满足更多专业学生的学习需求, 本书在借鉴国内外相关优秀教材的基础上, 着重突出三个特色. 第一是以精简、易懂、深入浅出的方式讲清楚基本概念、基本理论和推导技巧, 着重阐释统计思想和数据处理方法. 同时, 加强实用性, 通过大量实例, 一方面使得学习者深刻认识时间序列的基本概念、常用性质和基本理论; 另一方面也使得他们尽快掌握时间序列数据分析的基本技能. 第二是本书全程使用 Python 语言进行实例分析, 并且提供全部代码. Python 是一个结合了解释性、编译性、互动性和面向对象的高级程序设计语言. 它以“优雅、明确、简单”为设计哲学, 用它编写的程序简单易懂、易于维护, 具有很强的亲和力, 它免费开源, 具有很强的移植性、扩展性和嵌入性, 能够在各平台上顺利工作. 此外, Python 拥有丰富的数据分析库, 可适用于各种数据问题的处理, 大大节省了编写底层代码的时间. 尽管与 R 语言相比, Python 的统计模型没有那么多, 而且语法习惯相对不一致, 但是从其基本语法所产生的成千上万的模块使得它几乎可以做任何想做的事情. 因此, 近些年 Python 积累了大量的用户, 并已逐渐成为数据科学领域使用最广泛的语言之一. 第三是本书所使用的数据绝大多数是真实数据. 这些数据都可以在国家统计局网站、中国气象数据网等网站下载. 通过对真实数据的分析, 学习者更能体会到基本理论、数据分析技能和数据分析经验相结合的重要性. 同时, 也给初学者提供了大量免费获取数据资源和练习的机会. 读者可在各章节相应的地方扫二维码获取这些数据资源.

本书以时间序列分析的理论和实例相结合的方式, 有侧重地介绍了以下内容. 第 1 章概述时间序列的发展历程、时间序列的一些基本概念、数据建模的基本步骤和时间序列数据的预处理. 第 2 章和第 3 章分别介绍平稳时间序列模型的概念、性质、建模和预测方法.

第 4 章介绍时间序列数据分解的思想以及常用的数据平滑方法. 第 5 章介绍非平稳时间序列模型的概念、趋势的消除、ARIMA 模型的概念、性质、建模方法以及预测, 最后简单讨论了残差自回归模型. 第 6 章介绍几类常见的季节模型以及它们的建模和预测方法. 第 7 章讨论“伪回归”现象、单位根检验和协整. 第 8 章主要讲述 ARCH 模型和 GARCH 模型的概念、估计和检验. 第 9 章介绍时间序列谱分析的一些基本知识, 包括谱表示、谱密度及其估计. 第 10 章简要地介绍三种基于深度学习的时间序列预测方法, 主要包括基于多层感知机、循环神经网络和卷积神经网络的预测. 此外, 本书还配备了一定数量的习题. 目的是希望通过这些习题的演练, 使读者尽快掌握相应章节的基本理论和方法.

本书主要用作高等院校统计、经济、商科、工程以及定量社会科学等相关专业的高年级本科生学习时间序列分析的教材或教学参考书, 也可作为硕士研究生使用 Python 语言学习时间序列分析的入门书, 还可供相关技术人员进行时间序列数据处理时参考.

本书在写作过程中参考了国内外许多优秀的教材和论著, 在此向这些教材或著作的作者表示感谢和敬意. 本书能够及时出版, 还要感谢清华大学出版社刘颖编审的大力支持和帮助. 本书内容在大连民族大学统计学专业讲授多次, 感谢同学们对课程内容的浓厚兴趣和热烈讨论, 同时纠正了一些打印错误.

白晓东

2022 年 3 月

目　录

第 1 章　引言及基础知识

第1章数据资源

学习目标与要求

1. 了解时间序列分析的发展简史.
2. 理解时间序列的有关基本概念和主要特征.
3. 理解时间序列分析的基本步骤.
4. 学会时间序列数据预处理的方法.

1.1　引言

时间序列分析在人类早期的生产实践和科学研究中发挥了重要作用. 例如, 7000 年前, 古埃及人为了发展农业, 把尼罗河涨落的情况逐天记录下来, 并进行了长期的观察. 他们发现, 在天狼星第一次和太阳同时升起后的两百天左右尼罗河开始泛滥, 洪水持续七八十天, 此后土地肥沃、适于农业种植. 由于掌握了尼罗河泛滥的规律, 古埃及的农业迅速发展, 从而创造了古埃及灿烂的史前文明. 再如, 德国天文学家、药剂师 S. H. Schwabe (1789—1875) 从 1826 年至 1843 年, 在每一个晴天, 认真审视太阳表面, 并且记录下每一个黑点, 对这些记录仔细研究后, 最终发现了太阳黑子活动有 11 年左右的周期性规律. 这一发现被视为天文学上最重要的发现之一.

另外, 许多经济现象的发展都具有随时间演变的特征, 如宏观经济运行中的国内生产总值、消费支出、货币供应量等; 又如, 微观经济运行中的企业产品价格、销售量、销售额、利润等量; 再如, 金融市场中的股价指数、股票价格、成交量等变量的变化. 将这些变量依时间先后记录下来并加以研究, 揭示其中隐含的经济规律, 预测未来经济行为, 已经成为经济研究的重要手段.

像上面这样按照时间的顺序把随机事件变化发展的过程记录下来就构成了一个时间序列, 对时间序列进行观察、研究, 找寻它变化发展的规律, 预测它将来的走势就是时间序列分析.

1.1.1　时间序列的定义

在统计研究中, 一般将按时间顺序排列的一组随机变量

$$X_1, X_2, \cdots, X_t, \cdots \tag{1.1}$$

称为一个**时间序列 (time series)**, 简记为 $\{X_t, t \in T\}$ 或 $\{X_t\}$. 用

$$x_1, x_2, \cdots, x_n \tag{1.2}$$

或

$$\{x_t, t = 1, 2, \cdots, n\}$$

表示该随机序列的 n 个有序观察值, 称为**序列长度为 n 的观察值序列**, 有时也称观察值序列 (1.2) 为时间序列 (1.1) 的一个**实现**. 在上下文不引起歧义的情况下, 有时一个时间序列也记为 $\{x_t\}$.

下面介绍一些时间序列的例子.

例 1.1　把我国 1953 年至 2020 年国内生产总值 (gross domestic product, GDP) 按照时间顺序记录下来, 就构成了一个序列长度为 68 的国内生产总值观察值序列. 将数据按时间顺序逐一罗列或绘表罗列, 一般不易观察, 为此通常绘制时序图来观察趋势, 所谓时序图是指横轴表示时间, 纵轴表示时间序列的观察值而绘制的图. 借助 Python 绘图软件包可以绘制出许多漂亮的统计图.

首先, 导入 os 模块, 并用该模块中的 chdir() 函数改变工作路径; 导入 numpy 和 pandas 包, 并分别简记为 np 和 pd, 需要指出的是, 本例中没有用到 numpy 包; 导入 warnings 包, 并忽略一些非必要警告信息.

```
import os; os.chdir("D:\\TSBOOKDATA\\Chap1")      #改变工作目录
import numpy as np; import pandas as pd
import warnings; warnings.filterwarnings("ignore") #忽略警告信息
```

其次, 从绘图包 matplotlib 中导入模块 pyplot, 并简记为 plt. 具体语句如下:

```
%matplotlib inline                          #嵌入图形
from matplotlib import pyplot as plt
plt.rcParams['axes.unicode_minus']=False
plt.rcParams['font.family'] = "simsun"      #设置字体为 simsun
plt.style.use('ggplot')                     #设置背景样式
```

Matplotlib 是 Python 绘图的基石, 几乎所有与绘图有关的模块都会把它作为核心的底层. 在此基础上还有 seaborn、bokeh 等模块, 可以绘制风格更美观的图. Plotly 模块主要通过交互的方式来展现数据, 是绘图方面高级的模块. 读者可根据自身需要学习这些高级绘图模块.

需要说明的是, 由于以上两组语句在本书中反复使用, 因此为节省篇幅, 在本书之后的例子中, 我们总是将它们略去, 读者在运行程序时应该自行将它们加上. 这个说明也适用于其他的包、模块和函数. 另外, 如果在导入包或函数时出错, 那么应考虑是否已经安装了相应的包. 在本书中, 程序工作路径都默认为“D:/TSBOOKDATA/Chapx”(x 为章数), 读者在运行程序时应根据自己的实际情况, 进行修改.

再次, 读取数据, 并指定原数据的第一列作为显示索引, 然后作为 Series 赋值给 GDP. 具体语句如下:

```
GDP = pd.read_csv('ChinaGDP.csv',index_col=0,squeeze=True)
```

最后, 绘制时序图, 并将图像储存在当前目录下文件夹 fig 中, 储存格式为 png. 具体语句如下, 运行结果见图 1.1.

```
fig = plt.figure(figsize=(12,4), dpi=150)
ax = fig.add_subplot(111)
ax.plot(GDP, marker="o", linestyle="-", color='blue')
ax.set_ylabel(ylabel="国内生产总值(单位:亿元)", fontsize=17)
ax.set_xlabel(xlabel="年份", fontsize=17)
plt.xticks(fontsize=15); plt.yticks(fontsize=15)
fig.tight_layout(); plt.savefig(fname='fig/1_1.png')
```

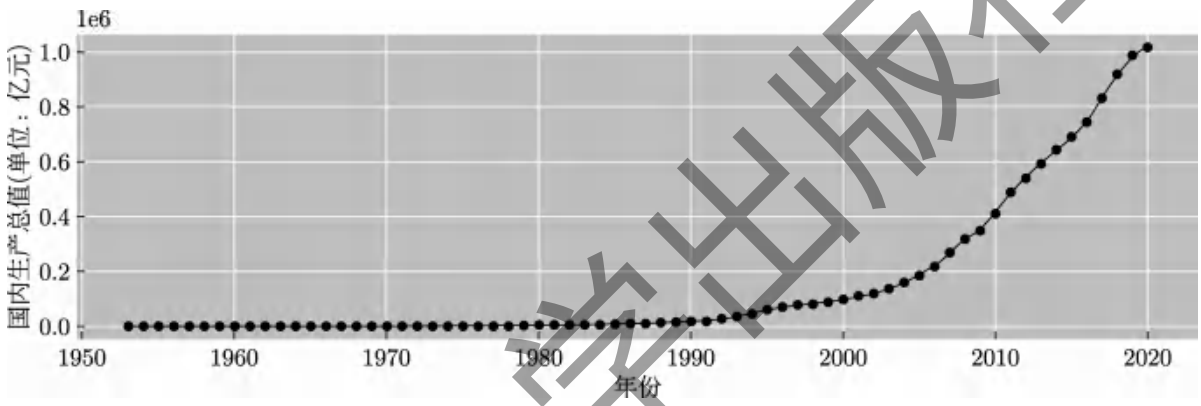

图 1.1 中国 1953 年至 2020 年年度国内生产总值的时序图

从图 1.1 中可以看出, 我国 GDP 从 1992 年开始大幅增长, 1998 年前后增长速度放缓, 而 2004 年之后, 除了 2009 年和 2020 年有小幅增速外, 几乎呈现直线型高速增长趋势.

为了更好地预测这种趋势, 我们关心的是相邻年度 GDP 的关联情况. 为此, 我们可以绘制我国当年 GDP 与上一年 GDP 的散点图. 接上面程序, 我们用下列 Python 语句生成图 1.2. 需要指出的是, 用 plt.show() 语句展示图片后, 需用 plt.close() 结束展示, 否则, 会占用内存资源. 在本书之后的绘图中, 为节省篇幅, 一般都会省去这两个函数, 读者在运行时, 要自己加上. 从图 1.2 可以看出, 相邻年度 GDP 的关联基本呈线性.

```
GDPy = GDP[1:]; GDPx = GDP[:-1]
fig = plt.figure(figsize=(12,4), dpi=150)
ax = fig.add_subplot(111)
ax.scatter(y=GDPy, x=GDPx, color='b', marker='o')
ax.set_ylabel(ylabel="当年 GDP", fontsize=17)
ax.set_xlabel(xlabel="上一年 GDP", fontsize=17)
plt.xticks(fontsize=15); plt.yticks(fontsize=15)
fig.tight_layout(); plt.savefig(fname='fig/1_2.png')
plt.show(); plt.close()
```

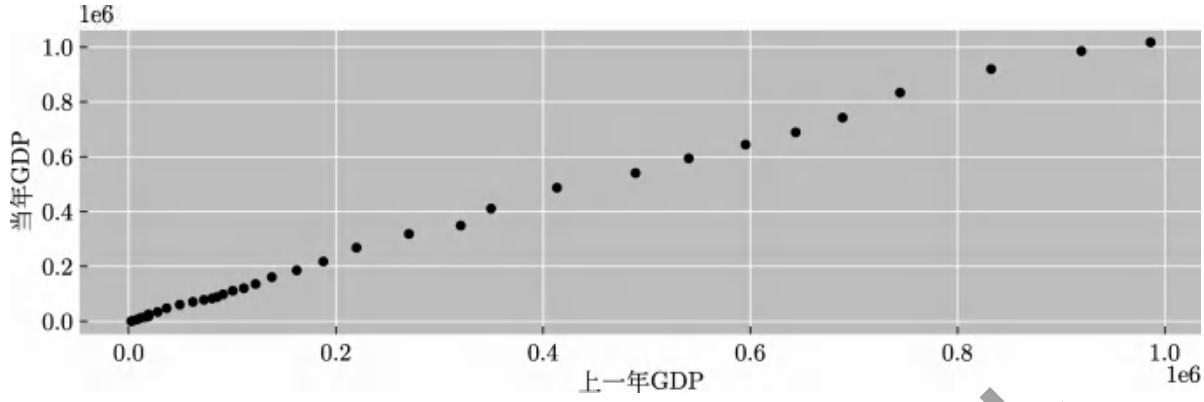

图 1.2　中国当年 GDP 与上一年 GDP 的散点图

例 1.2　将美国艾奥瓦州杜比克 (Dubic) 市从 1964 年 1 月至 1976 年 1 月这 144 个月的平均气温 (单位: 华氏度) 按时间顺序记录下来, 得到长度为 144 的观察值序列. 用下列 Python 语句生成图 1.3. 从图 1.3 可以看出, 这些观察值显示了很强的季节性趋势. 以后将会通过构造季节指数的方式, 对这类数据建模.

```
Dubic = np.loadtxt("DubicCity.txt")      #读入 .txt 格式数据
Index = pd.date_range(start="1964-01", end="1976-01", freq="M")
Dubic_ts = pd.Series(Dubic,index=Index) #建立时间序列
fig = plt.figure(figsize=(12,4),dpi=150)
ax = fig.add_subplot(111)
ax.plot(Dubic_ts,marker="o",linestyle="-",color='blue')
ax.set_ylabel(ylabel="气温",fontsize=17)
ax.set_xlabel(xlabel="年份",fontsize=17)
plt.xticks(fontsize=15); plt.yticks(fontsize=15)
fig.tight_layout(); plt.savefig(fname='fig/1_3.png')
```

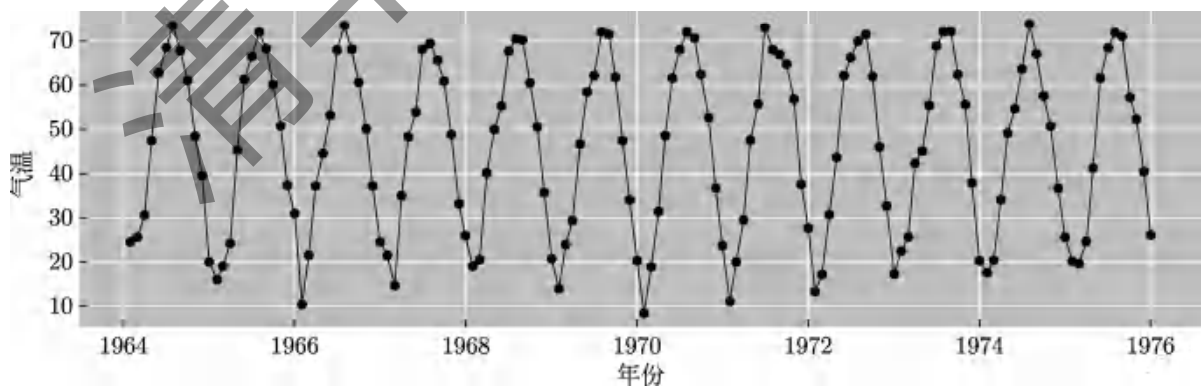

图 1.3　美国艾奥瓦州杜比克市月平均气温的时序图

例 1.3　将美国加利福尼亚州洛杉矶地区从 1880 年至 1995 年这 115 年来的年降水量记录下来, 构成一个序列长度为 115 的观察值序列. 用下列 Python 语句生成时序图 1.4.

```
LosAngeles = np.loadtxt("LosAngeles.txt")
LosTime = pd.date_range(start="1880", end="1995", freq="Y")
LosAngeles_ts = pd.Series(LosAngeles, index=LosTime)
fig = plt.figure(figsize=(12,4), dpi=150)
ax = fig.add_subplot(111)
ax.plot(LosAngeles_ts, marker="o", linestyle="-", color='blue')
ax.set_ylabel(ylabel="降水量 (单位: 英寸)", fontsize=17)
ax.set_xlabel(xlabel="年份", fontsize=17)
plt.xticks(fontsize=15); plt.yticks(fontsize=15)
fig.tight_layout(); plt.savefig(fname='fig/1_4.png')
```

从图 1.4 中可以看出, 该地区降水量没有明显的趋势性. 接下来观察相邻年份的相关关系. 由下列 Python 语句生成图 1.5. 从图 1.5 可以看出, 相邻各点没有明显的相关关系. 像这种既无明显的趋势, 也无明显的相关关系的数据, 从统计建模和预测角度看没有研究的意义.

```
Losx = LosAngeles_ts[1::]; Losy = LosAngeles_ts[:-1]
fig = plt.figure(figsize=(12,4), dpi=150)
ax = fig.add_subplot(111)
ax.scatter(y=Losy, x=Losx, color='b', marker='o')
ax.set_ylabel(ylabel="当年降水量 (单位:英寸)", fontsize=17)
ax.set_xlabel(xlabel="上一年降水量 (单位:英寸)", fontsize=17)
plt.xticks(fontsize=15); plt.yticks(fontsize=15)
fig.tight_layout(); plt.savefig(fname='fig/1_5.png')
```

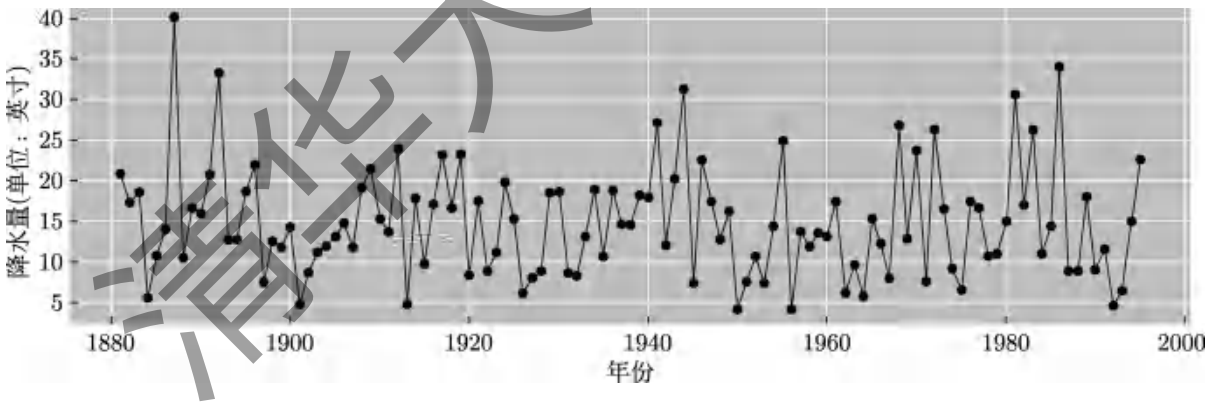

图 1.4 洛杉矶年降水量的时序图

从上述例子可以看出, 时间序列 (有时简称时序) 中观察值的取值随着时间的变化而不同, 反映了相关指标在不同时间进行观察所得到的结果. 这些观察值可以是一个时期内的数据, 也可能是一个时间点上的数据, 通常存在前后时间上的相依性. 从整体上看, 时间序列往往呈现某种趋势性或出现季节性变化的现象, 这种相依性就是系统的动态规律性, 也是进行时间序列分析的基础. 总之, 我们进行时间序列研究的目的是想揭示随机时序 $\{X_t\}$ 的性质, 而要实现这个目标就要分析它的观察值序列 $\{x_t\}$ 的性质, 由观察值序列的性质来建立恰当的模型, 从而推断随机时序 $\{X_t\}$ 的性质.

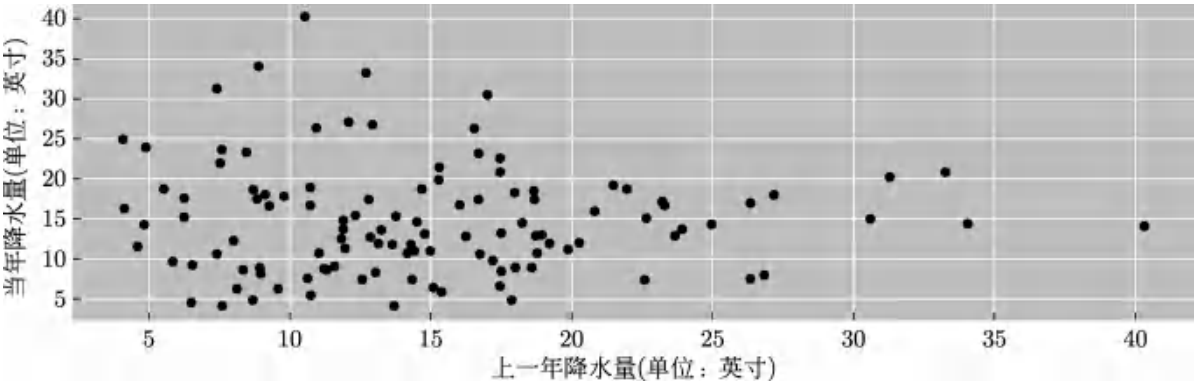

图 1.5　洛杉矶当年降水量与上一年降水量的散点图

1.1.2　时间序列的分类

在现实中存在不同类别的时间序列. 根据所研究问题的不同, 可以对时间序列做如下不同的分类.

1. 一元时间序列与多元时间序列

每个时间点只观察一个变量的时间序列称为**一元时间序列**. 如果每个时间点同时观察多个变量的时间序列则称为**多元时间序列**. 多元时间序列不仅描述了各个变量的变化情况, 而且还蕴含了各变量间的相互依存关系. 例如, 考查某国或某地区经济运行情况, 就需要同时观察该国国内生产总值、消费支出、投资额、货币供应量等一系列指标, 既要分析每个指标的动态变化情况, 还要分析各个指标之间的动态影响关系.

2. 连续时间序列与离散时间序列

时间序列是按照时间顺序记录的一系列观察值, 这种观察值可能是按连续的时间记录的, 也可能是按离散的时间点来记录的. 相应地, 通常把这两类序列分别称为**连续时间序列**和**离散时间序列**. 例如, 例 1.1～ 例 1.3 都是离散时间序列, 而利用脑电图记录仪记录大脑活动情况则可视为连续时间序列. 对于连续时间序列, 可通过等间隔抽取样本使之转化为离散时间序列加以研究. 一般地, 如果时间间隔足够小, 那么我们可以认为这种过程几乎不会损失原序列的信息.

3. 平稳时间序列与非平稳时间序列

按时间序列的统计性质, 可将时间序列分为**平稳时间序列**和**非平稳时间序列**. 关于时间序列的平稳性与非平稳性将在之后的学习中详细讨论.

此外, 还可以按照模型的表示形式分为**线性时间序列**和**非线性时间序列**, 等等.

1.1.3　时间序列分析的方法回顾

1. 描述性时间序列分析

早期的时间序列分析是通过直观的数据比较或绘图观测, 寻找序列中蕴含的发展规律, 这种分析方法就称为**描述性时间序列分析**. 该方法不采用复杂的模型和分析方法, 仅仅是按

照时间顺序收集数据, 描述和呈现序列的波动, 常常能使人们发现意想不到的规律, 具有操作简单、直观有效的特点. 人们在进行时间序列分析时, 往往首先进行描述性分析.

2. 统计时间序列分析

随着研究领域的不断拓展和深入, 单纯的描述性时间序列分析方法越来越显示出局限性. 在许多问题中, 随机变量的发展会表现出很强的随机性, 想通过对序列的简单观察和描述总结出随机变量发展变化的规律, 并准确预测出它们将来的走势通常非常困难. 为了准确地估计随机序列发展变化的规律, 从 20 世纪 20 年代开始, 学术界利用数理统计学原理分析时间序列值内在的相关关系, 由此开辟了一门应用统计学科 —— 时间序列分析.

从时间序列分析方法的发展历史来看, 其大致可分为两类：**频域 (frequency domain) 分析方法**和**时域 (time domain) 分析方法**.

频域分析方法也称为“频谱分析”或“谱分析”方法. 早期的频域分析方法假设任何一种无趋势的时间序列都可以分解成若干不同频率的周期波动, 借助 Fourier 分析从频率的角度揭示时间序列的规律. 20 世纪 60 年代, Burg 在分析地震信号时提出最大熵谱估计理论. 该理论克服了传统谱分析所固有的分辨率不高和频率泄露等缺点, 使得谱分析进入一个新的阶段, 称为现代谱分析. 谱分析方法是一种非常有用的纵向数据分析方法, 目前已广泛应用于物理学、天文学、海洋学、气候学、电力和通信工程等领域. 谱分析最大的缺点是, 需要较强的数学基础才能熟练使用, 而且分析结果较为抽象, 难以解释.

时域分析方法的基本思想是事件的发展通常都具有一定的惯性, 这种惯性用统计学语言来描述就是序列值之间存在一定的相关关系, 而且这种相关关系具有某种统计规律性. 我们分析的重点就是从序列自相关的角度揭示时间序列的某种统计规律. 相对于谱分析方法, 它具有理论基础扎实、操作步骤规范、分析结果易于解释等优点. 目前已经广泛应用于自然科学和社会科学的各个领域, 成为时间序列分析的主流方法之一.

时域分析方法的起源可以追溯到 20 世纪 20 年代英国统计学家 G. U. Yule 在分析和预测市场变化规律等问题中提出的自回归模型 (AR 模型). 同时, 英国科学家 G. T. Walker 爵士在研究气象问题时得到著名的 Yule-Walker 方程. 这些开创性工作奠定了时域分析方法的基础. 20 世纪 60 年代之后, 随着计算机技术和数据处理技术的迅速发展, 时间序列分析的理论和应用得到迅猛发展. 1970 年, 统计学家 G. E. P. Box 和 G. M. Jenkins 在梳理、发展已有研究成果的基础上, 共同撰写了 *Time Series Analysis: Forecasting and Control* 一书. 该书系统地阐述了对求和自回归移动平均模型 (ARIMA 模型) 的识别、估计、检验和预测的原理和方法. 这些方法已经成为经典的时域分析方法.

在此基础上, 人们不断拓展研究方法. 20 世纪 80 年代以来, 统计学家逐步转向多变量场合、异方差场合和非线性场合的时间序列分析方法的研究, 并取得突破性的进展. 1982 年, R. F. Engle 在研究英国通货膨胀率的建模问题时, 提出了自回归条件异方差模型 (ARCH 模型). 而 Bollerslov 在 1985 年提出的广义自回归条件异方差模型 (GARCH 模型) 则进一步放宽了自回归条件异方差模型的约束条件. 之后, Nelson 等又提出指数广义自回归条件异方差模型 (EGARCH 模型)、方差无穷广义自回归条件异方差模型 (IGARCH 模型) 和依均值广义自回归条件异方差模型 (GARCH-M 模型) 等限制条件更为宽松的异方差模型, 大大推广和补充了自回归条件异方差模型. 它们比传统的方差齐性模型更准确地刻画了金融市场风险的变化过程. R. F. Engle 因此获得 2003 年的诺贝尔经济学奖. 在多变量方面, C. Granger 于

1987 年提出协整理论, 极大地促进了多变量时间序列分析方法的发展. C. Granger 也于 2003 年获得诺贝尔经济学奖. 在非线性场合, 各种新的模型纷纷被提出. Granger 和 Anderson 在 1978 年提出双线性模型; 汤家豪于 1989 年提出门限自回归模型; Chen 和 Tasy 于 1993 年提出非线性可加模型, 等等. 非线性模型是个异常广阔的研究领域, 在该领域中, 模型构造、参数估计、参数检验等各方面都有大量的研究工作需要探究.

1.2　基本概念

在本节中, 我们介绍一些时间序列分析过程中的基本概念. 这些基本概念表明了本书中所研究的时间序列的主要统计性质.

1.2.1　时间序列与随机过程

我们知道, 随机变量是分析随机现象的重要工具, 对于简单的随机现象, 用一个随机变量就可以了, 如某时段内共享单车的使用量, 某时刻候车的人数, 等等. 而对于复杂的随机现象, 用一个随机变量描述就不够了, 需要用若干个随机变量来描述. 一般地, 将一族随机变量放在一起就构成一个随机过程. 具体地, 有下面的定义: 我们将概率空间 $(\Omega, \mathcal{F}, P)$ 上的一族随机变量 $\{X_t, t \in T\}$ 称为一个**随机过程 (stochastic process)**, 其中 t 是参数, 它属于某个集合 T, 通常称 T 为**参数集 (parameter set)**.

参数集 T 可以是离散集, 也可以是连续集. 若 T 为一连续集, 则 $\{X_t\}$ 为一连续型随机过程. 若 T 为离散集, 则称 $\{X_t\}$ 为一离散型随机过程. 当参数集为某时间集合时, 则相应的随机过程就是时间序列. 可见, 时间序列仅仅是随机过程的特殊情况, 因此随机过程的许多概念和性质同样适用于时间序列.

1.2.2　概率分布族及其特征

由数理统计的知识可知, 分布函数能够完整地描述一个随机变量的统计性质. 同样, 要刻画时间序列的统计特征, 就要探讨一列随机变量的统计分布.

设 $\{X_t, t \in T\}$ 为一个随机过程, 对于任意一个 $t \in T$, X_t 是一个随机变量, 它的分布函数 $F_{X_t}(x)$ 可以通过 $F_{X_t}(x) = P(X_t \leqslant x)$ 得到, 这一分布称为**时间序列的一维分布**. 对于 $t_1, t_2 \in T$, 有两个随机变量 X_{t_1}, X_{t_2} 与之对应, X_{t_1}, X_{t_2} 的联合分布函数为 $F_{X_{t_1}, X_{t_2}}(x_1, x_2) = P(X_{t_1} \leqslant x_1, X_{t_2} \leqslant x_2)$, 称为**时间序列的二维联合分布**.

一般地, 任取正整数 n 以及 $t_1, t_2, \cdots, t_n \in T$, 则 n 维向量 $(X_{t_1}, X_{t_2}, \cdots, X_{t_n})^{\mathrm{T}}$ 的联合分布函数为

$$F_{X_{t_1}, X_{t_2}, \cdots, X_{t_n}}(x_1, x_2, \cdots, x_n) = P(X_{t_1} \leqslant x_1, X_{t_2} \leqslant x_2, \cdots, X_{t_n} \leqslant x_n).$$

这些有限维分布函数的全体

$$\{F_{X_{t_1},X_{t_2},\cdots,X_{t_n}}(x_1,x_2,\cdots,x_n),\forall n\in\mathbf{Z}^+,\forall t_1,t_2,\cdots,t_n\in T\}$$

称为**时间序列 $\{X_t,t\in T\}$ 的有限维分布族**.

理论上, 时间序列 $\{X_t,t\in T\}$ 的所有统计性质都可通过有限维分布族推导出来, 但是在实际应用中, 要想得到一个时间序列的有限维分布族几乎是不可能的, 而且有限维分布族在使用中通常涉及非常复杂的数学运算, 因而一般情况下, 我们很少直接使用有限维分布族进行时间序列的分析. 事实上, 在时间序列分析中, 更简单实用的方法是通过数字特征来研究其统计规律. 常用的关于时间序列的数字特征有如下几种.

1. 均值函数

对时间序列 $\{X_t,t\in T\}$ 来说, 任意时刻的序列值 X_t 都是一个随机变量. 假设它的分布函数为 $F_{X_t}(x)$, 那么当

$$\mu_t=E(X_t)=\int_{-\infty}^{+\infty}x\mathrm{d}F_{X_t}(x)<+\infty$$

对于所有 $t\in T$ 成立时, 我们称 μ_t 为时间序列 $\{X_t,t\in T\}$ 的**均值函数 (mean function)**. 它反应的是时间序列 $\{X_t,t\in T\}$ 在各个时刻的平均取值水平.

2. 方差函数

当对于所有 $t\in T$

$$\int_{-\infty}^{+\infty}x^2\mathrm{d}F_{X_t}(x)<+\infty$$

成立时, 我们称

$$\sigma_t^2=\mathrm{Var}(X_t)=E(X_t-\mu_t)^2=\int_{-\infty}^{+\infty}(x-\mu_t)^2\mathrm{d}F_{X_t}(x)$$

为时间序列 $\{X_t,t\in T\}$ 的**方差函数 (variance function)**. 它反应了序列值围绕其均值做随机波动时的平均波动程度.

3. 自协方差函数

类似于随机变量间的协方差, 在时间序列分析中, 我们可以定义**自协方差函数 (autocovariance function)** 的概念. 对于时间序列 $\{X_t,t\in T\}$, 任取 $t,s\in T$, 我们称

$$\gamma(t,s)=\mathrm{Cov}(X_t,X_s)=E[(X_t-\mu_t)(X_s-\mu_s)]$$

为序列 $\{X_t,t\in T\}$ 的自协方差函数.

4. 自相关函数

同样地, 类似于随机变量间的相关系数, 我们可以定义时间序列的**自相关函数 (auto-**

correlation function, ACF). 我们称

$$\rho(t,s)=\mathrm{Cor}(X_t,X_s)=\frac{\gamma(t,s)}{\sqrt{\mathrm{Var}(X_t)}\sqrt{\mathrm{Var}(X_s)}}$$

为序列 $\{X_t,t\in T\}$ 的自相关函数. 时间序列的自协方差函数和自相关函数反应了不同时刻的两个随机变量的相关程度.

5. 偏自相关函数

自相关函数虽然反应了时间序列 $\{X_t,t\in T\}$ 在两个不同时刻 X_t 和 X_s 的相依程度, 但是这种相关包含了 X_s 通过 X_t 和 X_s 之间的其他变量 $X_{s+1},X_{s+2},\cdots,X_{t-1}$ 传递到对 X_t $(s<t)$ 的影响, 也就是说自相关函数实际上掺杂了其他变量的影响. 为了剔除中间变量的影响, 人们引入**偏自相关函数 (partial autocorrelation function, PACF)** 的概念. 偏自相关函数定义为

$$\beta(s,t)=\mathrm{Cor}(X_t,X_s|X_{s+1},\cdots,X_{t-1})=\frac{\mathrm{Cov}(X_t,X_s|X_{s+1},\cdots,X_{t-1})}{\sqrt{\mathrm{Var}(X_t)}\sqrt{\mathrm{Var}(X_s)}},\quad 0<s<t.$$

一般来讲, 一个时间序列的上述数字特征与时间有关, 因而可看成关于时间的函数. 不同类型时间序列的数字特征会随时间变化呈现不同的变化规律, 如有些时间序列的均值函数或方差函数不随时间的变化而变化, 有些时间序列的自相关函数或偏自相关函数会出现随时间推移而逐渐变小的规律, 等等. 在之后的章节中, 我们将详细讨论不同类型时间序列在数字特征中表现出的差异.

1.2.3　平稳时间序列的定义

对时间序列进行统计推断时, 通常要对其做出某些简化的假设, 其中最重要的假设是平稳性. 根据限制条件的严格程度, 时间序列的平稳性可分为严平稳和宽平稳两个层面.

1. 严平稳时间序列

严平稳是一种条件较为严格的平稳性定义, 它要求序列的所有有限维分布不随时间的推移而发生变化, 从而序列的全部统计性质也不会随着时间的推移而发生变化. 具体地, 定义如下：

设 $\{X_t,t\in T\}$ 为一时间序列. 若对于任意正整数 n, 任取 $t_1,t_2,\cdots,t_n\in T$ 以及任意正数 h, 都有

$$F_{X_{t_1+h},X_{t_2+h},\cdots,X_{t_n+h}}(x_1,x_2,\cdots,x_n)=F_{X_{t_1},X_{t_2},\cdots,X_{t_n}}(x_1,x_2,\cdots,x_n),$$

则称时间序列 $\{X_t,t\in T\}$ 为**严平稳时间序列 (strictly stationary time series)**.

严平稳时间序列的定义所要求的条件过分严格. 实际中, 要想知道时间序列 $\{X_t,t\in T\}$ 的有限维分布族是极其困难的事情, 而在此基础上判断一个时间序列是否属于严平稳则更难, 所幸时间序列的主要统计性质是由它的低阶矩决定的, 因此可以把严平稳的条件放宽, 仅仅要求其数字特征不随时间发生变化, 这样就得到了宽平稳的概念.

2. 宽平稳时间序列

一般地, 如果一个时间序列 $\{X_t, t \in T\}$ 满足如下三个条件：

(1) 对于任意的 $t \in T$, 有 $E(X_t) = \mu$, μ 为常数;

(2) 对于任意的 $t \in T$, 有 $E(X_t^2) < +\infty$;

(3) 对于任意的 $s, t, k \in T$, 且 $k + t - s \in T$ 有

$$\gamma(s, t) = \gamma(k, k + t - s), \quad 0 < s < t.$$

则称 $\{X_t, t \in T\}$ 为**宽平稳时间序列 (weakly stationary time series)**. 宽平稳也称为弱平稳或二阶矩平稳.

宽平稳的条件显然比严平稳的条件宽泛得多, 更具有可操作性, 它只要求二阶矩具有平稳性, 二阶以上的矩没有做任何要求. 一般情况下, 宽平稳不一定是严平稳; 严平稳也不一定是宽平稳, 如服从柯西分布的严平稳序列就不是宽平稳序列, 因为它不存在一、二阶矩, 所以无法验证它二阶矩平稳. 不过, 存在二阶矩的严平稳序列一定是宽平稳的. 宽平稳一般推不出严平稳, 但当序列服从多元正态分布时, 由宽平稳可以推出严平稳.

例 1.4 如果一个时间序列 $\{X_t, t \in T\}$ 满足: 任取正整数 n 和任意的 $t_1, t_2, \cdots, t_n \in T$, 相应的 n 维随机变量 $\boldsymbol{X}_n = (X_{t_1}, X_{t_2}, \cdots, X_{t_n})^{\mathrm{T}}$ 服从 n 维正态分布, 密度函数为

$$f_{\boldsymbol{X}_n}(\boldsymbol{x}_n) = (2\pi)^{-\frac{n}{2}} |\boldsymbol{\Gamma}_n|^{-\frac{1}{2}} \exp\Big[-\frac{1}{2}(\boldsymbol{x}_n - \boldsymbol{\mu}_n)^{\mathrm{T}} \boldsymbol{\Gamma}_n^{-1} (\boldsymbol{x}_n - \boldsymbol{\mu}_n)\Big],$$

其中, $\boldsymbol{x}_n = (x_{t_1}, x_{t_2}, \cdots, x_{t_n})^{\mathrm{T}}$; $\boldsymbol{\mu}_n = (\mu_{t_1}, \mu_{t_2}, \cdots, \mu_{t_n})^{\mathrm{T}}$; T 为向量或矩阵的转置符号; $\boldsymbol{\Gamma}_n$ 为协方差阵:

$$\boldsymbol{\Gamma}_n = \begin{pmatrix} \gamma(t_1, t_1) & \gamma(t_1, t_2) & \cdots & \gamma(t_1, t_n) \\ \gamma(t_2, t_1) & \gamma(t_2, t_2) & \cdots & \gamma(t_2, t_n) \\ \vdots & \vdots & & \vdots \\ \gamma(t_n, t_1) & \gamma(t_n, t_2) & \cdots & \gamma(t_n, t_n) \end{pmatrix},$$

那么我们称其为**正态时间序列**.

从正态随机序列的密度函数可以看出, 它的 n 维分布仅由均值向量和协方差阵决定, 因此对于正态随机序列而言, 宽平稳一定严平稳.

需要强调的是, 在实际应用中, 如果不做说明, 我们所说的平稳指的就是宽平稳.

1.2.4 平稳时间序列的一些性质

根据平稳时间序列的定义, 可以将自协方差函数由二维函数 $\gamma(t, s)$ 简化为一维函数 $\gamma(s - t)$：

$$\gamma(t - s) \overset{\mathrm{def}}{=} \gamma(s, t), \quad \forall t, s \in T,\ t > s.$$

由此得到延迟 k 自协方差函数的概念.

一般地, 对于平稳时间序列 $\{X_t, t \in T\}$, 称

$$\gamma(k) = \gamma(t, t+k), \quad \forall t, t+k \in T$$

为该时间序列的**延迟 k 自协方差函数**.

根据平稳时间序列的定义可知, 平稳序列具有常数方差

$$\mathrm{Var}(X_t) = \gamma(t,t) = \gamma(0), \quad \forall t \in T.$$

由延迟 k 自协方差函数的概念可以等价得到**延迟 k 自相关函数**的概念, 为

$$\rho(k) = \frac{\gamma(t, t+k)}{\sqrt{\mathrm{Var}(X_t)}\sqrt{\mathrm{Var}(X_{t+k})}} = \frac{\gamma(k)}{\gamma(0)}.$$

容易验证延迟 k 自相关函数具有如下三个性质:

(1) 规范性

$$\rho(0) = 1 \quad \text{且} \quad |\rho(k)| \leqslant 1, \quad \forall k.$$

(2) 对称性

$$\rho(k) = \rho(-k).$$

(3) 非负定性

根据协方差阵的非负定性, 可得对于任意正整数 m, 相关矩阵

$$\boldsymbol{P}_m = \begin{pmatrix} \rho(0) & \rho(1) & \cdots & \rho(m-1) \\ \rho(1) & \rho(0) & \cdots & \rho(m-2) \\ \vdots & \vdots & & \vdots \\ \rho(m-1) & \rho(m-2) & \cdots & \rho(0) \end{pmatrix}$$

为非负定矩阵.

我们应注意的是, 虽然一个平稳时间序列唯一决定了它的自相关函数, 但是一个自相关函数未必唯一对应一个平稳时间序列, 因而延迟 k 自相关函数 $\rho(k)$ 对应的模型并不唯一. 这个性质给我们根据样本自相关函数来确定模型增加了难度. 在后面的章节我们将进一步说明这个问题.

1.2.5　平稳性假设的意义

数理统计学是利用样本信息来推测总体信息, 时间序列分析作为数理统计学的一个分

支也不例外。根据统计学常识, 要分析一个 n 维随机向量 $\boldsymbol{X}=(X_1,X_2,\cdots,X_n)^{\mathrm{T}}$, 需要如表 1.1 中结构的数据.

表 1.1 数据表

样本	随机变量		
	X_1	$\cdots$	X_n
1	x_{11}	$\cdots$	x_{n1}
2	x_{12}	$\cdots$	x_{n2}
$\vdots$	$\vdots$		$\vdots$
m	x_{1m}	$\cdots$	x_{nm}

显然, 我们希望维数 n 越小越好, 而对于每个变量希望样本容量 m 越大越好, 这是因为维数越小分析过程越简单, 样本容量越大, 分析结果越可靠. 但是对于时间序列而言, 它在任意时刻 t 的序列值都是一个随机变量, 而且由于时间的不可重复性, 该变量在任意一个时刻只能获得唯一的样本观察值, 其数据结构如表 1.2 所示. 由于某时刻对应的随机变量的样本容量太小, 用该数据直接分析此刻的随机变量基本不会得到可用的结果, 因此必须借用一些辅助信息, 才能得到一些有用的结果. 序列平稳性假设是解决该问题的有效途径之一.

表 1.2 数据表

样本	随机变量			
	X_1	$\cdots$	X_t	$\cdots$
1	x_1	$\cdots$	x_t	$\cdots$

如果一个时间序列是平稳的, 那么其均值函数是常数, 也即 $\{\mu_t,\ t\in T\}$ 变成了常数序列 $\{\mu,\ t\in T\}$。这样, 本来每个随机变量 X_t 的均值 μ_t 只能凭借唯一的样本观察值 x_t 来估计, 即 $\hat{\mu}=x_t$, 现在由于 $\mu_t\equiv\mu,\forall t\in T$, 于是每个样本观察值 $x_t,\forall t\in T$ 都变成了 μ 的样本观察值

$$\hat{\mu}=\overline{x}=\frac{1}{n}\sum_{i=1}^{n}x_i.$$

于是, 不但提高了对均值函数的估计精度, 而且大大降低了时间序列分析的难度。

同样地, 基于平稳性可计算出延迟 k 自协方差函数的估计值

$$\hat{\gamma}(k)=\frac{1}{n-k}\sum_{t=1}^{n-k}(x_t-\overline{x})(x_{t+k}-\overline{x})$$

和总体方差的估计值

$$\hat{\gamma}(0)=\frac{1}{n-1}\sum_{t=1}^{n}(x_t-\overline{x})^2.$$

进而可得延迟 k 自相关函数的估计值

$$\hat{\rho}(k)=\frac{\hat{\gamma}(k)}{\hat{\gamma}(0)},\quad \forall 0<k<n.$$

当延迟阶数 k 远远小于样本容量时, 有

$$\hat{\rho}(k) \approx \frac{\sum_{t=1}^{n-k}(x_t - \overline{x})(x_{t+k} - \overline{x})}{\sum_{t=1}^{n}(x_t - \overline{x})^2}, \quad \forall 0 < k < n.$$

1.3　时间序列建模的基本步骤

从实际数据出发, 对时间序列建模一般可遵循四个步骤, 即模型识别、模型估计、模型检验和模型应用. 通常上述四个步骤需要经过多次反复, 才能达到比较满意的效果.

1.3.1　模型识别

从实际数据出发建立时间序列模型时, 首先就要进行模型识别. 所谓**模型识别**就是根据时间序列的统计特征选择适当的拟合模型. 通俗地讲, 就是根据数据的特征, 判断所研究的时间序列属于哪一类. 模型识别主要包含如下内容:

(1) 依照所研究的问题科学地收集数据.

(2) 根据时间序列的数据作出相关图, 求出相关函数进行分析. 相关图能够显示出序列变化的趋势性和周期性等特征, 这些特征不但隐含着序列的平稳性的一些特点, 而且能够发现跳点和拐点. 而这些跳点和拐点也是模型识别的重要参考因素.

(3) 判别时间序列是平稳的还是非平稳的. 一般来讲, 判别时间序列的平稳性有两种方法, 一种是图检验法; 另一种是构造统计量进行假设检验的方法. **图检验法**是根据时序图和自相关图显示的特征做出平稳性判别的方法. 它的优点是操作简便、运用广泛; 它的缺点是判别结论带有很强的主观色彩, 因此最好能够用统计检验方法加以辅助判别. 目前最常用的平稳性统计检验方法是**单位根检验**.

(4) 判别时间序列是否为纯随机序列. 当对一个时间序列进行了平稳性判别之后, 序列被分成了平稳序列和非平稳序列两类. 对于非平稳序列通常要通过进一步的检验、变换或处理, 才能够确定适当的拟合模型. 对于平稳序列来讲, 我们需要检验其是否为纯随机的, 因为只有那些序列值之间具有密切相关关系的序列, 才值得我们花时间去挖掘历史数据中的有效信息, 用来预测序列未来的发展. 如果序列值彼此之间没有任何相关性, 那就意味着该序列是一个没有记忆的序列, 过去的行为对将来的发展没有丝毫影响, 这种序列称为**纯随机序列**. 从统计分析的角度而言, 纯随机序列没有任何分析的价值.

(5) 综合考虑时间序列的统计特征辨识合适的模型类型, 初步确定模型结构.

至于常见的时间序列模型有哪些, 它们分别具有哪些统计特征, 以及如何根据样本信息估计数字特征、识别拟合模型等, 将在后续章节详细研究.

1.3.2 模型估计

依照样本信息进行模型识别之后, 我们得到了所分析的时间序列大概服从什么样的模型类型和模型结构, 模型的最终形式还需要估计模型的参数之后才能够确定. 模型的参数决定了不同时刻随机变量之间的相依关系, 也即反映了随机变量随时间变化的记忆性大小和记忆期的长短. 当参数确定了, 变量的动态关系也就确定了. 比如, 通过模型识别判断出时间序列 $\{X_t\}$ 服从二阶自回归模型 (AR(2))

$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + \varepsilon_t,$$

其中 $\{\varepsilon_t\}$ 是均值为零的白噪声序列 (见 35 页). 模型中的参数 ϕ_1, ϕ_2 表明了 $\{X_t\}$ 的当前值对其前两个时刻的值的依赖程度, 或者说记忆的大小.

在数理统计中, 估计时间序列模型参数的常用方法有: 矩估计、极大似然估计和最小二乘估计. **矩估计方法**是用样本矩代替相应的总体矩, 并通过求解相应的方程而得到参数估计的方法; **极大似然估计**是使得样本出现概率最大, 也就是使得似然函数达到最大而得到参数估计的方法; **最小二乘估计**是使得模型拟合的残差平方和达到最小, 从而求得参数估计的方法. 这三种方法都有各自的优点和不足. 矩估计方法具有简单、直观和计算量小等优点; 其缺点是, 利用信息不足, 估计效率低以及估计精度不高等. 一般进行时间序列分析时, 先用矩估计方法进行初步估计, 然后使用极大似然估计方法或者非线性最小二乘方法进行精确估计.

1.3.3 模型检验

在模型识别时, 为了简化问题我们会提出一些假设, 这些假设往往因人而异, 带有主观因素, 因此必须对模型本身进行检验. 同时, 由于参数估计方法本身也有许多缺点, 而且有些参数贡献不大, 甚至可以忽略, 所以对所估出的参数也必须进行检验. 由于上述两个原因, 所以时间序列模型的检验有两类, 一类是模型的显著性检验; 另一类是模型参数的显著性检验. 这两类检验统称为模型的**诊断性检验**.

模型的显著性检验主要是检验模型的有效性. 一个模型是否有效主要看它提取的相关信息是否充分, 一个好的拟合模型应该确保提取出了观察值序列中几乎所有的样本相关信息, 换言之, 拟合残差项中将不再蕴含任何相关信息, 即残差序列应该为白噪声序列 (其概念见后续章节). 反之, 如果残差序列为非白噪声序列, 那就意味着残差序列中还残留着相关信息未被提取, 这就说明拟合模型不够有效, 需重新选择模型进行拟合.

模型参数的显著性检验主要是检验模型中每一个参数是否显著异于零. 目的是要找出贡献不大的参数并将其剔除, 使得模型更为精简和准确. 一般地, 如果模型中包含了不显著的参数, 不但使得模型参数冗余, 影响自由度, 而且也会影响其他参数的估计精度.

在实际应用中, 如果模型的诊断性检验没有通过, 则需要重新识别、估计和检验, 直到得到一个满意的拟合模型.

如果一个模型通过了检验, 说明在一定的置信水平下, 该模型能够有效地拟合观察值序列的波动, 但这种有效模型有时并不是唯一的. 面对多个显著有效的模型, 到底选择哪个来统

计推断更好呢? 为了解决这个问题, 一般需要引进一些信息准则来进行模型优化. 具体地, 在后继章节中, 我们结合具体模型来详细论述.

1.3.4　模型应用

时间序列模型的应用主要包括变量动态结构分析、预测和控制.

动态结构分析是指用已经估计出参数的模型, 对变量的动态变化情况进行考查. 例如, 对于自回归模型 (AR 模型), 可以考查它的记忆特征和记忆衰减情况; 对于滑动平均模型 (MA 模型), 可以考查外部冲击对变量的影响情况和对外部冲击的记忆期限. 动态结构分析对于认识经济金融变量的运行规律具有重要作用.

预测是时间序列建模的最重要的目的, 是指用已经估计出参数的模型, 对变量未来变化进行预报. **控制**是指根据时间序列模型调整输入变量使得系统发展过程保持在目标值上. 当运用时间序列模型进行预测、发现预测值会偏离目标值时, 便可进行必要的控制, 调整当前值使之朝预定目标靠近.

总之, 时间序列建模过程包括模型识别、模型估计、模型检验和优化, 并可能反复多次才能达到比较满意的效果, 最终投入使用. 时间序列分析完整的流程可用图 1.6 表示.

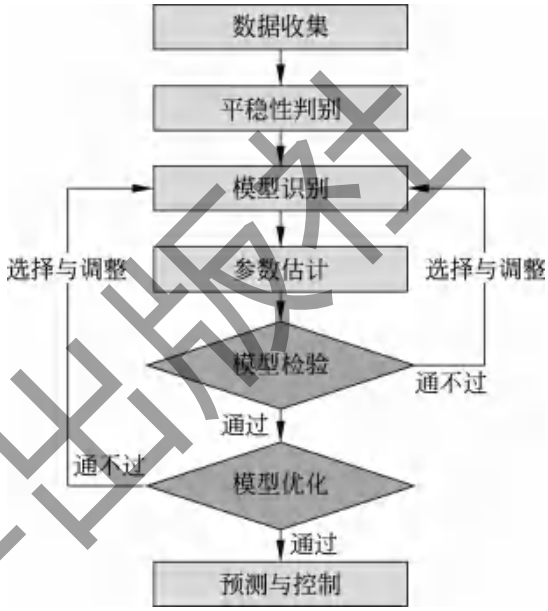

图 1.6　时间序列分析流程图

1.4　数据预处理

时间序列数据建模之初, 我们应该对数据有个初步的认识, 如: 大致观察一下数据的趋势性、季节性; 序列值相近时期的相关性; 初步判断一下序列的平稳性; 判断序列是否有研究的必要, 即检验一下序列是否是白噪声. 这些数据建模前的分析都称为数据的预处理.

1.4.1　时序图与自相关图的绘制

进行时间序列分析的第一步, 通常是利用序列值画出时序图和自相关图进行观察. 正如前面所定义, **时序图**就是一张二维平面图, 一般横坐标表示时间, 纵坐标表示序列取值. 而所谓**自相关图**是平面上的悬垂线图, 横坐标表示延迟时期数, 纵坐标表示自相关系数. 悬垂线表示自相关系数的大小. 通过观察时序图, 我们能够获得序列值的趋势和走向; 通过自相关图我们能够大致获得不同时刻序列值之间的相关关系. 这些能够帮助我们初步判断序列的统计特性.

借助于 Python 语言强大的绘图功能可绘制出所需要的序列时序图和自相关图. 下面举例说明时序图和自相关图的绘制方法.

例 1.5 现给出 2000 年 1 月至 2012 年 10 月新西兰人出国旅游目的地的数据. 下面我们一起来认识这组时间序列数据, 并用 Python 绘制时序图来分析.

首先, 我们读取数据的前两行来认识所给数据的结构, 然后再决定用哪些数据和 read_csv() 中的哪些参数. 具体命令及运行结果如下:

```
nzt = pd.read_csv("NZTravellersDestination.csv")
nzt.head(2)
运行结果:
Date       Australia  Fiji   China   India   UK    US
2000/01    23203      1936   2285    2270    5418  4109
2000/02    16546      1052   1485    1539    3023  3520
```

运行结果显示, 数据分别统计了新西兰人到 Australia、Fiji、China、India、UK、US 这些国家的旅游人数.

其次, 我们绘制 2000 年 1 月至 2012 年 10 月新西兰人月均来中国旅游人数的时序图. 具体命令如下, 运行结果见图 1.7.

```
NtoC = pd.read_csv('NZTravellersDestination.csv', usecols=
       ['Date','China'], parse_dates=['Date'], index_col='Date')
fig = plt.figure(figsize=(12,4), dpi=150)
ax = fig.add_subplot(111)
ax.plot(NtoC, marker="o", linestyle="-", color='blue')
ax.set_ylabel(ylabel="人数", fontsize=17)
ax.set_xlabel(xlabel="年份", fontsize=17)
plt.xticks(fontsize=15); plt.yticks(fontsize=15)
fig.tight_layout(); plt.savefig(fname='fig/1_7.png')
```

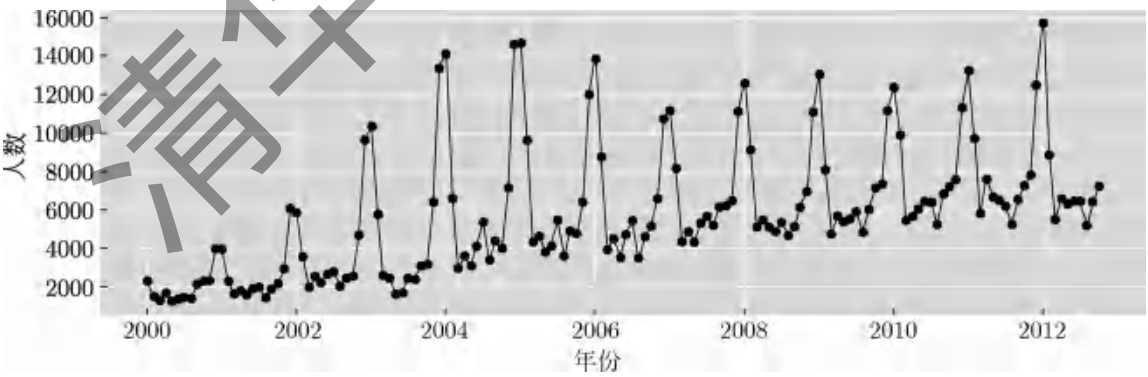

图 1.7 2000 年 1 月至 2012 年 10 月新西兰人月均来中国旅游的时序图

从时序图 1.7, 我们可以看到从 2000 年 1 月到 2012 年 10 月新西兰人月均来中国旅游人数有增长趋势但是增幅和增速不大. 同时新西兰人月均来中国旅游人数有明显的季节性. 每年的圣诞节前后来华旅游人数最多.

我们也可以在同一个窗口, 绘制不同变量的时序图, 来比较这些变量之间的变化. 下面比

较新西兰人从 2000 年 1 月至 2012 年 10 月月均到中国、印度、英国和美国四国旅游人数变化的情况. 具体命令如下, 运行结果见图 1.8.

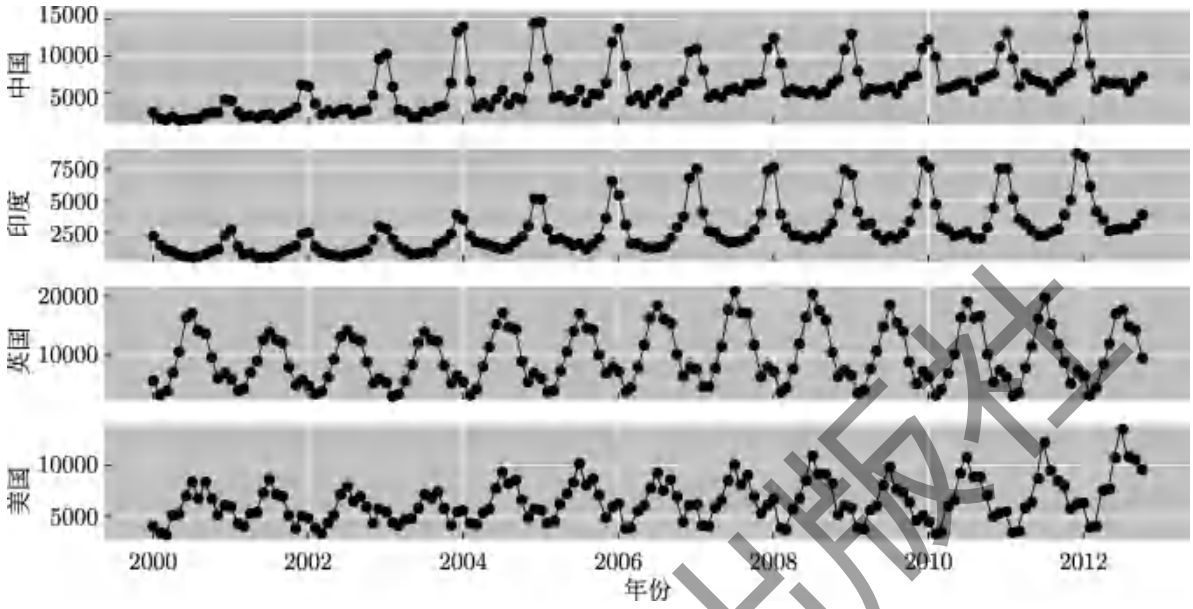

图 1.8　2000 年 1 月至 2012 年 10 月新西兰人月均出国旅游的时序图

```
ucls = ['Date','China','India','UK','US']
NtoF = pd.read_csv('NZTravellersDestination.csv', usecols=ucls,
       parse_dates=['Date'], index_col='Date')
Date = NtoF.index; NtoC = NtoF.China.values; NtoUS = NtoF.US.values
NtoI = NtoF.India.values; NtoU = NtoF.UK.values
new,(ax1,ax2,ax3,ax4) = plt.subplots(4, 1, sharex=True,
                        figsize=(12,6), dpi=150)
ax1.plot(Date,NtoC, color='r',marker="o")
ax1.set_ylabel(ylabel="中国", fontsize=17)
ax2.plot(Date,NtoI, color='b',marker="o")
ax2.set_ylabel(ylabel="印度", fontsize=17)
ax3.plot(Date,NtoU, color='y', marker="o")
ax3.set_ylabel(ylabel="英国", fontsize=17)
ax4.plot(Date,NtoUS, color='g',marker="o")
ax4.set_ylabel(ylabel="美国", fontsize=17)
ax4.set_xlabel(xlabel="年份", fontsize=17)
plt.xticks(fontsize=15); plt.yticks(fontsize=15)
fig.tight_layout(); plt.savefig(fname='fig/1_8.png')
```

语句说明: 第一句用所需数据的列名建立了一个列表. 第二句读取文件中所需的列, 并把 Date 列作为时间索引. 第三、四句从数据框 NtoF 中提取出新西兰人分别到中国、美国、印度和英国旅游的月度数据. 第五句将画布分成四个子图, 并共用 x 轴. 第六句至第十六句绘图并调整横纵轴标签. 最后一句将绘制完成的图命名保存.

从图 1.8 可以看出, 2000 年 1 月至 2012 年 10 月新西兰人月均到中国和印度的人数都有增长趋势, 来中国旅游的人数增长更快些. 去英国和美国的人数比较稳定. 另外, 这四组数据都有明显的季节性. 在每年 12 月左右到中国、印度旅游的人数最多; 而在夏季新西兰人更乐意到英国和美国旅游.

在绘图时, 为了突出比较效果, 可以使用 hlines() 和 vlines() 函数为图形添加水平和垂直参照线. 下面通过例子来说明这两个函数的使用.

例 1.6　绘制 2018 年 10 月至 2021 年 9 月北京市商品住宅施工面积累计值 (单位: 万平方米) 的时序图, 并添加辅助线来比较. 具体命令如下, 运行结果见图 1.9.

```
import matplotlib.ticker as ticker
bjch = pd.read_csv('BJCH.csv', encoding = 'utf-8')
for f in bjch:                          #线性插值
    bjch[f] = bjch[f].interpolate()
    bjch.dropna(inplace=True)
xlabs = bjch.Time; ticker_spacing = xlabs; ticker_spacing = 5
fig = plt.figure(figsize=(12,4), dpi=150)
ax = fig.add_subplot(111)
ax.plot(xlabs, bjch.CCA, color='b', marker='o')
ax.xaxis.set_major_locator(ticker.MultipleLocator(ticker_spacing))
ax.vlines(x=['2020/04','2020/09'], ymin=5230, ymax=6250,
          color="g", linestyle='--')
ax.hlines(y=[5250,6250], xmin='2020/04', xmax='2020/09',
          color="g", linestyle='--')
ax.set_xlabel(xlabel='时间', fontsize=17)
ax.set_ylabel(ylabel='施工面积累计值', fontsize=17)
plt.xticks(fontsize=15); plt.yticks(fontsize=15)
fig.tight_layout(); plt.savefig(fname='fig/1_9.png')
```

语句说明: 第一句导入 ticker 库, 以便改变数据轴的间距来解决日期显示太密的问题. 第三句至第五句进行线性插值, 填充缺失值. 倒数第七句设置数据轴日期的显示.

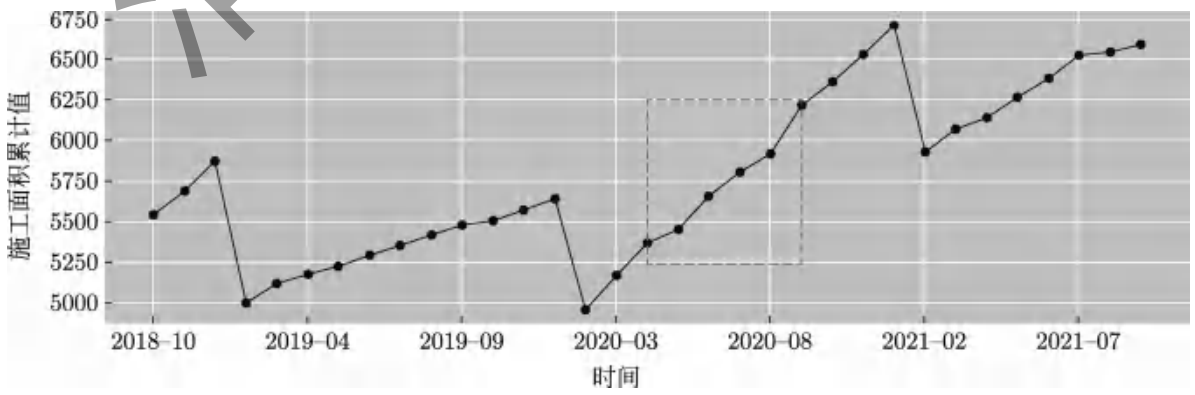

图 1.9　2018 年 10 月至 2021 年 9 月北京市商品住宅施工面积累计值的时序图

例 1.7　接例 1.6, 绘制 2018 年 10 月至 2021 年 9 月北京市商品住宅施工面积累计值的自相关图.

首先, 导入 stattools 模块中的自相关函数 acf(). 自相关函数 acf(ts, nlags) 可以用来计算序列 ts 的延迟 nlags 阶数的自相关系数.

```
from statsmodels.tsa.stattools import acf
```

其次, 为了今后使用方便, 我们定义了一个绘制自相关函数图的函数 ACF():

```
def ACF(ts, lag=20, fname=" "):
    lag_acf = acf(ts, nlags=lag, fft=False)
    plt.vlines(x=list(range(lag+1)), ymin=np.zeros(lag+1),
               ymax=lag_acf, linewidth=2.0, color='black')
    plt.axhline(y=0, linestyle=':', color='blue')
    plt.axhline(y=-1.96/np.sqrt(len(ts)), linestyle='--',
                color='red')
    plt.axhline(y=1.96/np.sqrt(len(ts)), linestyle='--',
                color='red')
    plt.title(''); plt.xticks(fontsize=15); plt.yticks(fontsize=15)
    plt.xlabel(xlabel="lag", fontsize=17)
    plt.ylabel(ylabel="ACF", fontsize=17)
    plt.tight_layout(); plt.savefig(fname=fname)
```

最后, 调用自定义函数 ACF() 绘制自相关函数图, 运行结果见图 1.10.

```
fig = plt.figure(figsize=(12,4), dpi=150)
ax = fig.add_subplot(111)
ACF(bjch['CCA'], lag=30, fname="fig/1_10.png")
```

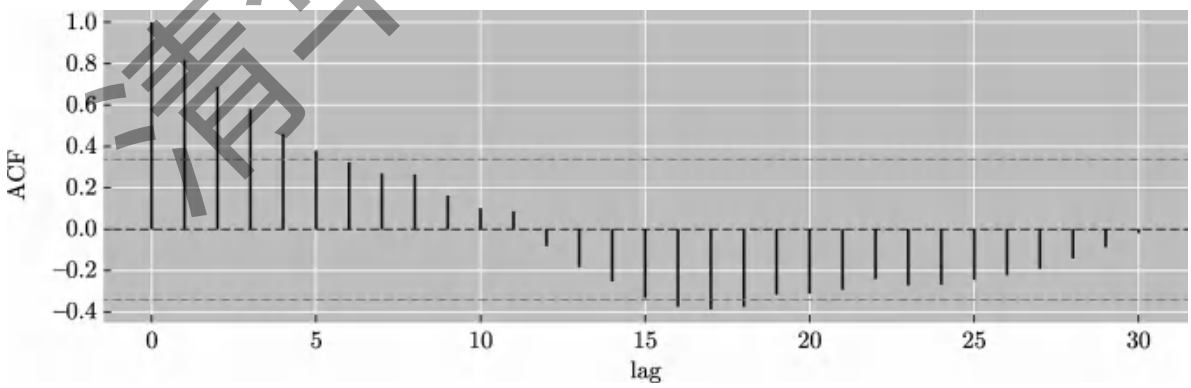

图 1.10　商品住宅施工面积累计值的自相关图

图 1.10 中的水平虚线表示正态分布 $N(0,1/n)$ 的 95% 的置信区间 $[-1.96/\sqrt{n}, 1.96/\sqrt{n}\,]$. 一般地, 当自相关函数的悬垂线落在两条虚线围成的区域之外时, 说明相应序列值之间具有显著相关性, 而当自相关函数的悬垂线落入这两条虚线围成的区域之内时, 则说明相应序列