

探索性数据分析

(西北大学 夏志明)

完成日期: December 23, 2003

目 录

I	数据分析原理与数据分析系统	1
第一章	数据分析原理与策略	2
1.1	Historical developments in data analysis	2
1.1.1	DDA、IDA and EDA	2
1.1.2	Rao's Data-analysis Procedures	5
1.2	开放系统与探索性数据分析的进一步拓展	5
1.2.1	开放系统与数据挖掘	5
1.2.2	数据挖掘流程	5
第二章	常见的数据分析系统	6
2.1	SAS系统	6
2.2	SPSS系统	6
2.3	Minitab系统	6
2.4	R	6
2.5	Matlab	6
II	SAS系统与数据分析	1
第三章	SAS语言初步	2
3.1	SAS语言概述	2
3.1.1	一个SAS程序	2
3.1.2	SAS程序基本结构与基本语法	3
3.1.3	SAS程序的生命历程	9
3.2	SAS表达式	11
3.2.1	数据集与数据库	11
3.2.2	常量	13
3.2.3	变量	14
3.2.4	SAS函数	14
3.2.5	运算符与运算类型	19
3.2.6	表达式	21

3.3	数据集与矩阵	21
3.3.1	数据集及其查询与运算 (SQL过程)	21
3.3.2	矩阵运算与IML过程	21
3.4	Data步机理与常见语句	22
3.4.1	Data步语言概述与内循环(Built-in Loop)	22
3.4.2	Data步标志语句	25
3.4.3	Data步可执行语句 (一) ——基础数据操作 (行列运算)	26
3.4.4	Data步可执行语句 (二) ——数据集操作(表运算)	32
3.4.5	Data步数组语句	38
3.4.6	Data步控制语句	42
3.5	Proc步机理、功能与常见语句	49
3.5.1	Proc步机理与功能	49
3.5.2	Proc步通用语句	49
3.6	常见全程语句	53
3.6.1	概述	53
3.6.2	注释语句	53
3.6.3	TITLE语句	54
3.6.4	OPTIONS语句	54
3.6.5	X语句	54
3.6.6	FOOTNOTE语句	55
3.6.7	RUN语句	55
3.6.8	LIBNAME语句	56
3.6.9	FILENAME语句	57
3.6.10	MISSING语句	58
3.7	文件输出系统—ODS	58
3.8	宏语言—宏变量与宏	58
3.8.1	宏机制 (Macro Facility)	58
3.8.2	宏变量	60
3.8.3	宏	65
3.8.4	宏变量、常量、宏函数与宏表达式	70
3.8.5	宏语句与宏控制结构	70
3.8.6	Data步接口程序	71
3.8.7	宏语言的应用实例	73

第 四 章	数据集的建立	74
4.1	方法概述与实例分析	74
4.1.1	方法概述	74
4.1.2	实例分析	74
4.2	程序实现方法	75
4.2.1	Data步数据集的建立机制	75
4.2.2	数据集的建立	75
4.3	菜单实现方法	77
4.3.1	子系统输入	77
4.3.2	外部文件转换	79
第 五 章	抽样调查	82
5.1	抽样调查概述	82
5.1.1	(直接)数据收集的方法—抽样调查与非抽样调查	82
5.1.2	总体与样本	83
5.1.3	总体特征与估计量	86
5.1.4	误差与精度	88
5.1.5	抽样调查的一般流程	89
5.2	常见抽样方法	90
5.2.1	简单随机抽样	90
5.2.2	分层抽样	91
5.2.3	整群抽样	92
5.2.4	二阶段与多阶段抽样	92
5.2.5	系统抽样(等距抽样)	93
5.2.6	不等概率抽样	94
5.3	SAS抽样过程—surveyselect	94
5.3.1	surveyselect过程及其它随机数理论	94
5.3.2	抽样方法的SAS实现	95
第 六 章	简单统计分析—参数估计与假设检验	103
6.1	假设检验与SAS过程	103
6.1.1	假设检验基础	103
6.1.2	单样本检验问题	105
6.1.3	两样本检验问题	111
6.2	相关分析与CORR过程	115

第七章 线性回归分析	116
7.1 统计方法与实例分析	116
7.1.1 多元线性回归分析基础	116
7.1.2 过程介绍	126
7.2 程序实现方法	126
7.2.1 基本多元回归分析	126
7.2.2 基本诊断分析	129
7.2.3 变量选择	132
7.3 菜单实现方法	135
7.3.1 基本多元回归分析	135
7.3.2 基本诊断分析	136
7.3.3 变量选择	137
第八章 方差分析模型(ANOVA)与试验设计问题	140
8.1 方差分析引论	140
8.1.1 理论模型的建立	140
8.1.2 新药的临床试验	144
8.2 单因子试验	148
8.2.1 单因子试验的统计模型	148
8.2.2 固定效应模型的统计分析	149
8.2.3 随机效应模型的统计分析	152
8.2.4 模型诊断	152
8.3 多因子试验	152
8.3.1 两因子试验的统计模型	152
8.3.2 固定效应模型的统计分析	152
8.3.3 随机效应模型与混合模型的统计分析	152
8.3.4 多因子试验的设计与分析	152
8.3.5 拉丁方设计与正交拉丁方设计	152
8.4 试验设计中的面板模型	152
8.4.1 面板模型概述	152
8.4.2 临床试验中的交叉试验	152
8.4.3 重复测量设计	152

第 九 章 经典时序分析	153
9.1 统计方法与实例分析	153
9.2 程序实现方法	153
9.3 菜单实现方法	153
第 十 章 Logistic回归与广义线性模型	154
10.1 Logistic模型及其统计推断	154
10.2 流行病学中的统计模型基础	154
10.3 Logistic回归程序实现方法	154
10.4 Logistic回归菜单实现方法	154
10.5 广义线性模型	154
第 十 一 章 多元数据处理	155
11.1 聚类分析及其实现	156
11.1.1 统计方法与实例分析	156
11.1.2 程序实现方法	156
11.1.3 菜单实现方法	156
11.2 判别分析及其实现	156
11.2.1 统计方法与实例分析	156
11.2.2 程序实现方法	156
11.2.3 菜单实现方法	156
11.3 主成分分析及其实现	156
11.3.1 统计方法与实例分析	156
11.3.2 程序实现方法	156
11.3.3 菜单实现方法	156
11.4 因子分析及其实现	156
11.4.1 统计方法与实例分析	156
11.4.2 程序实现方法	156
11.4.3 菜单实现方法	156
11.5 相关分析及其实现	156
11.5.1 统计方法与实例分析	156
11.5.2 程序实现方法	156
11.5.3 菜单实现方法	156

III SPSS系统与数据分析	157
第十二章 SPSS语言介绍与系统简介	158
第十三章 多元统计分析	159
13.1 回归分析及其实现	160
13.1.1 统计方法与实例分析	160
13.1.2 程序实现方法	160
13.1.3 菜单实现方法	160
13.2 主成分分析及其实现	160
13.2.1 主成分分析	160
13.2.2 实例分析与菜单实现方法	160
13.3 因子分析及其实现	160
13.3.1 因子分析	160
13.3.2 实例分析与菜单实现方法	160
13.4 聚类分析及其实现	160
13.4.1 统计方法与实例分析	160
13.4.2 程序实现方法	160
13.4.3 菜单实现方法	160
13.5 判别分析及其实现	160
13.5.1 统计方法与实例分析	160
13.5.2 程序实现方法	160
13.5.3 菜单实现方法	160
13.6 典型相关分析及其实现	160
13.6.1 统计方法与实例分析	160
13.6.2 程序实现方法	160
13.6.3 菜单实现方法	160
13.7 对应分析及其实现	160
13.7.1 统计方法与实例分析	160
13.7.2 程序实现方法	160
13.7.3 菜单实现方法	160
13.8 多维标度分析及其实现	160
13.8.1 统计方法与实例分析	160
13.8.2 程序实现方法	160
13.8.3 菜单实现方法	160

13.9 广义线性模型及其实现	160
13.9.1 统计方法与实例分析	160
13.9.2 程序实现方法	160
13.9.3 菜单实现方法	160
13.10 对数线性模型及其实现	160
13.10.1 统计方法与实例分析	160
13.10.2 程序实现方法	160
13.10.3 菜单实现方法	160
13.11 广义判别分析及其实现	160
13.11.1 统计方法与实例分析	160
13.11.2 程序实现方法	160
13.11.3 菜单实现方法	160
13.12 生存分析及其实现	160
13.12.1 统计方法与实例分析	160
13.12.2 程序实现方法	160
13.12.3 菜单实现方法	160
 IV Minitab系统与数据分析	 161
 第十四章 Minitab系统介绍	 162
 第十五章 Minitab功能介绍	 163
 参考文献	 164

第一部分 I

数据分析原理与数据分析系统

第一章

数据分析原理与策略

§1.1 Historical developments in data analysis

*“Data! data! he cried impatiently,
I can’t make bricks without clay.”*

Conan Doyle —The Copper Beeches

```
proc means data=incomes mean var std cv skewness  
alpha=0.1 kurtosis t prt clm maxdec=2;  
var income;  
run;
```

§1.1.1 DDA、IDA and EDA

Styles in statistical analysis change over time while the object of “extracting all the information from data” or “summarization and exposure” remains the same. Statistics has not yet aged into a stable discipline with complete agreement on foundations. Certain methods become popular at one time and are replaced in course of time by others which look more fashionable. In spite of controversies, the statistical methodology and fields of applications are expanding. The computers together with the availability of graphic facilities have had a great impact on data analysis. It may be of interest to briefly review some historical developments in data analysis.

It has been customary to consider descriptive and theoretical statistics as two branches of statistics with distinct methodologies. In the former, the object is to summarize a given data set in terms of certain “descriptive statistics” such as measures of location and dispersion, higher order moments and indices, and also to exhibit salient features of the data through graphs such as histograms, bar diagrams, box plots and two dimensional charts. No reference is made to the stochastic mechanism (or probability distribution) which gave rise to the observed data. The descriptive statistics thus computed are used to compare different data sets. Even some rules are prescribed for the choice among alternative statistics, such as the mean, median and mode, depending on the nature of the data set and the questions to be answered. Such statistical analysis is referred to as descriptive data analysis (DDA). In theoretical statistics, the object is again summarization of data, but with reference to a specified family (or model) of underlying probability distributions. The summary or descriptive statistics in such a case heavily depend on the specified stochastic model, and their distributions are used to specify margins of uncertainty in inference about the unknown parameters. Such methodology is referred to as inferential data analysis (IDA).

Karl Pearson (K.P.) was the first to try to bridge the gap between DDA and IDA. He used the insight provided by the descriptive analysis based on moments and histograms to draw inference on the underlying family of distributions. For this purpose he invented the first and perhaps the most important test criterion, the chi-squared statistic, to test the hypothesis that a given data arose from a specified stochastic model (family of probability distributions) or consistent with a given hypothesis, which "ushered in a new sort of decision making" [See Hacking (1984), where K.P.'s chi-squared is eulogized as one of the top 20 discoveries' since 1900 considering all branches of science and technology. Even R.A. Fisher (R.A.F.) who had personal differences with K.P. expressed his appreciation of K.P.'s chi-squared test in personal conversation with the author.] K.P. also created a variety of probability distributions distinguishable by four moments. A beautiful piece of research work done by K.P. through the use of histograms and chi-squared test is the discovery that the distribution of the size of trypanosomes found in certain animals is a mixture of two normal distributions (see Pearson (1914- 15)).

The need to develop general methods of estimation arose in applying the chi-squared test to examine a composite hypothesis that the underlying distribution belongs to a specified parametric family of distributions. K.P. proposed the estimation of parameters by moments, and using the chi-squared test based on the fitted distribution. Certain refinements were made by R.A.F. both in terms of obtaining a better fit to given data through the estimation of unknown parameters by the method of maximum likelihood and also in the exact use of the chi-squared test using the concept of degrees of freedom when the unknown parameters are estimated.

During the twenties and thirties, R.A.F. created an extraordinarily rich array of statistical ideas. In a fundamental paper in 1922 he laid the foundations of "theoretical statistics," of analyzing data through specified stochastic models. He developed exact small sample tests for a variety of hypotheses under normality assumption and advocated their use with the help of tables of certain critical values, usually 5 period, under the influence of R.A.F., great emphasis was laid on tests of significance and numerous contributions were made by Hotelling, Bose, Roy and Wilks among others to exact sampling theory. Although R. A.F. mentioned specification, the problem first considered by K.P., as an important aspect of statistics in his 1922 paper, he did not pursue the problem further. Perhaps in the context of small data sets arising in biological research which R.A.F. was examining, there was not much scope for investigating the problem of specification or subjecting observed data to detailed descriptive analysis to look for special features or to empirically determine suitable transformations of data to conform to an assumed stochastic model. R.A.F. used his own experience and external information of how data are ascertained in deciding on specification. [See the classical paper by R.A.F. (1934) on the effect of methods of ascertainment on the estimation of frequencies.] At this stage of statistical developments inspired by R.A.F.'s approach, attempts were made by others to look for what are called nonparametric test criteria whose distributions are independent of the underlying stochastic model for the data (Pitman, 1937) and to investigate robustness of test criteria proposed by R.A.F. for departures from normality of the underlying distribution.

The twenties and thirties also saw systematic developments in data collection through design of experiments introduced by R.A.F., which enabled data to be analyzed in a specified manner through analysis of variance and interpreted in a meaningful way: design dictated the analysis and analysis revealed the design.

While much of the research in statistics in the early stages was motivated by problems arising in biology, parallel developments were taking place in a small scale on the use of statistics in industrial production. Shewhart (1931) introduced simple graphical procedures through control charts for detecting changes in a production process, which is probably the first methodological contribution to detection of outliers or change points.

Much of the methodology proposed by R.A.F. was based on intuition, and no systematic theory of statistical inference was available except for some basic ideas in the theory of estimation. R.A.F. introduced the concepts of consistency, efficiency and sufficiency and the method of maximum likelihood in estimation. J. Neyman and E.S. Pearson in 1928 (see their collected papers) provided some kind of axiomatic setup for deriving appropriate statistical methods, especially in testing of hypotheses, which was further pursued and perfected by Wald (1950) as a theory for decision making. R.A.F. maintained that his methodology was more appropriate in scientific inference while conceding that the ideas of Neyman and Wald might be more relevant in technological applications, although the latter claimed universal validity for their theories. Wald also introduced sequential methods for application in sampling inspection, which R.A.F. thought had applications in biology also. [In an address delivered at the ISI, R.A.F. mentioned, Shewhart's control charts, Wald's sequential sampling and sample surveys as three important developments in statistical methodology.]

The forties saw the development of sample surveys which involved collection of vast amounts of data by investigators by eliciting information from randomly chosen individuals on a set of questions. In such a situation, problems such as ensuring accuracy (free from bias, recording and response errors) and comparability (between investigators and methods of enquiry) of data assumed paramount importance. Mahalanobis (1931, 1944) was perhaps the first to recognize that such errors in survey work were inevitable and could be more serious than sampling errors, and steps should be taken to control and detect these errors in designing a survey and to develop suitable scrutiny programs for detecting gross errors (outliers) and inconsistent values in collected data.

We have briefly discussed what are commonly believed to be two branches of statistics, viz., descriptive and inferential statistics, and the need felt by practicing statisticians to clean up the data of possible defects which may vitiate inferences drawn from statistical analysis. What was perhaps needed is an integrated approach, providing methods for a proper understanding of given data, its defects and special features, and for selection of a suitable stochastic model or a class of models for analysis of data to answer specific questions and to raise new questions for further investigation. A great step in this direction was made by Tukey (1962, 1977) and Mosteller and Tukey (1968) in developing what is known as exploratory data analysis (EDA). The basic philosophy of EDA is to understand the special features of data and to use robust procedures to accommodate for a wide class of possible stochastic models for the data. Instead of asking the Fisherian question as to what

summary statistics are appropriate for a specified stochastic model, Tukey proposed asking for what class of stochastic models, a given summary statistic is appropriate. Reference may also be made to what Chatfield (1985) describes as initial data analysis, which appears to be an extended descriptive data analysis and inference based on common sense and experience with minimal use of traditional statistical methodology.

§1.1.2 Rao's Data-analysis Procedures

§1.2 开放系统与探索性数据分析的进一步拓展

§1.2.1 开放系统与数据挖掘

一 生命活力与开放系统

唯物史观：世界是物质的且相互联系；物质是运动变化的，运动有其基本规律。唯物史观产生自然辩证法，特别是系统论，即：任何对象和过程都以开放系统的形式存在和发展。较完备的开放系统能越来越发展，而较不完备的开放系统则渐趋灭亡。

唯物史观认为人的认识过程须过程一个较为完备的开放系统，如此才能越来越有生命力；人的行为模式也应该是一个开放系统，否则会偏离客观。

从认识论的角度看，数据分析可以视为一个从实践到理论的认识过程，或者是一个思维的归纳过程，因而必须形成一个开放系统。图基的探索性数据分析的方法具有由数据到结论的开放系统思想。也成为现代统计学家进行数据分析越来越青睐的原因。

二 两种建模思想——数据建模与算法建模

需要仔细思考“探索性数据分析思想”的一个重要表现——算法建模的思想，并结合Bruce, Hansen最新的两篇“平均思想”进行变点检验估计和检验的文章，仔细体会和思考算法建模的思想的合理性及其将对经典统计学的冲击和深远影响。

§1.2.2 数据挖掘流程

数据挖掘的基本流程的要点包括：

1)。数据收集

2)。结合先验知识和需解决的问题，对数据进行前期处理

3)。建立模型与统计诊断，注意到这是一个小的交互圈。统计诊断基于模型，同时也从诊断信息也能改进模型；特别的，从诊断能逃出该模型，为进一步的模型提供一个交互圈；甚至更进一步为数据收集与前期处理提供基础。

4)。模型评价

该步是基于较为合适的模型，也就是说，在“建模”与“诊断”相纠结的小交互圈，能顺利解脱的条件下，模型可以进行评价。或者，另外一种观点，模型评价可以独立出来，具有不依赖于模型的某种独立性。

5)。模型应用

第二章

常见的数据分析系统

§2.1 SAS系统

§2.2 SPSS系统

§2.3 Minitab系统

§2.4 R

§2.5 Matlab

第二部分 II

SAS系统与数据分析

第三章

SAS语言初步

§3.1 SAS语言概述

§3.1.1 一个SAS程序

为了整体印象，先给出一个简单的SAS程序，如下：

SAS源码 3.1: Create a dataset and summary it

```
1 Data zhili ;
2 input name $ x1 x2 x3;
3 cards;
4 Zhang 14 13 28
5 Li 10 14 15
6 Wang 11 12 19
7 Zhao 7 7 7
8 Wu 13 12 24
9 Liu 19 14 22
10 Zhong 20 16 26
11 Yang 9 10 14
12 Zeng 9 8 25
13 He 9 9 12
14 ;
15
16 Proc means data=zhili;
17 var x1 x2 x3;
18 run;
```

上述代码短小，但给出的就是一个完整的程序，所谓“麻雀虽小，五脏俱全”，下面从不同角度审视一个SAS程序的特点。

比较系统的审视，是分层审视，大概有三层：“语汇”、“语句”和“篇章”。最低层次是“语汇”，语汇是SAS最基本的语义单位。例如：上述程序代码中，都是语汇，语汇是不可再分的，在这个例子中，我们看到：

1).关键词：data, input, cards, proc, var, run是关键词；means是过程名；data=是means语句中的选项；

2).标识符：zhili是数据集名；name, x1, x2, x3是变量名；

3).特殊字符：“\$”是特殊字符，标志变量属性；“；”也是特殊字符，标志语句的结束。

当然如果从句子的角度看,可以将句子从不同角度进行分类,例如按是否产生计算机操作,可以区分为“说明语句”、“控制语句”和“可执行语句”;按语句的作用范围,可执行语句区分为“局部语句”和“全程语句”等等,不一而足。

最后,由句子可以构成篇章。篇章的结构认识,对于梳理和学习整个SAS语言,甚至一般的计算机语言都是很关键的,一般可以从两个角度认识:一个角度是从承担固定功能的语句块区分;另一个角度就是从程序的输入、处理以及输出三大逻辑块来认识。而这两者恰好存在某种程度的对应关系。

从第一个角度而言,SAS程序包括三块:数据步,过程步以及其他语句区。数据步和过程步是SAS的基本语句块,前者主要为了建立数据集(当然也可以包括一些简单的数据处理工作,如:后面要讲到的数据变换、描述性统计量以及利用SAS函数做的更复杂的数据操作),后者主要为了数据处理。其他语句区包括三类语句:宏语句,ODS语句以及其他全程语句。这种认识很大程度是从语句的空间分布,或者也可以说从语句的作用范围而言的。

SAS程序的另一个认识角度就是从逻辑块角度:数据输入、数据处理和数据输出。数据处理是程序的灵魂,对于SAS这种进行统计分析的软件而言,数据处理更为重要。

1). 语言的基本风格—数据运算、数据处理及其控制结构。

基本而灵活的数据处理主要要体现在各种数据结构的运算。需要注意的是,这些计算机语言的基本风格有一部分由数据步完成,如:变量的基本运算,数据集的常见运算,基本上由Base模块支持;有一部分由过程步完成,如:矩阵运算以及两者的矩阵与数据集的相互转换由IML过程完成,这个过程又由IML模块支持。最后,运算的基本控制结构,基本上由Base模块支持。

2). SAS语言的模块结构特色—数据步、过程步与全程语句。

数据步主要完成数据的输入工作(建立数据集)以及简单的数据处理;高级但不够灵活的数据分析技巧则主要体现在一些统计过程中;全程语句是两模块之外的一些语句。三者构成了SAS语言的一种空间分布特色,是SAS独有的。SAS的这种简单的空间模块特色,承载着复杂多变的逻辑功能,包括上述基本的数据运算与控制、高级的数据处理与控制(宏机制)以及下述的数据输出(ODS)系统。空间模块与逻辑功能大体对应,并不严格,例如:数据步完成数据输入和简单处理,过程完成数据的复杂处理,以及数据输出,注意到这是不严格的,如:数据步也有输出,过程步也有输入(IML过程的矩阵建立)。

3). SAS语言的输出风格—ODS系统。

最后输出工作体现在ODS上,这是一个贯穿数据步、过程步以及其他部分的系统。

4). SAS语言的全局控制—宏机制。

宏机制的出现,很好的协调了这几部分工作,使得这几部分工作具有更高的效率。

§3.1.2 SAS程序基本结构与基本语法

SAS程序分为三个层次认识:语汇、句子和篇章,分别有对应的一般的游戏规则,他们构成一般的语法内容。然而除了,一般的语法之外,每个具体的句子承载者具体的任务,也有具体的要求,这些特殊的句子或者过程的规则需要具体去理解和记忆,特别是强调,具体的语法需要结合数据分析的任务和基本步骤进行记忆,这是做好SAS语言的基本功。下面所讲是一般语法基础。

一 语汇与词法

定义 3.1 语汇是SAS程序的基本要素，是最基本的语义单位，包括关键词（保留字）、标识符、运算符和SAS语言允许使用的其他特殊字符。

☞ 几点说明 ☞：

✓ 语汇包括四类：关键词，标识符，运算符，特殊字符。四类语汇不重合。
 ✓ 标识符是可以按某些规则唯一自主命名的语汇，在风格严谨的计算机程序中难得的彰显个性色彩。常见的标识符有两类

- 1). 变量名、数组名、矩阵名、数据集名、数据库名
- 2). 函数名

其实这两类的区别在于：前者标志某种数据结构。后者则可以从两个角度理解，一是函数描述了一个过程，从该角度，有专家将函数视为一种变换或者运算，似乎更为合理，而函数名只是标志了这么一个过程；另一方面，因为函数均有返回值，此时函数名标志的就是该可变的返回值，从该角度也可以把函数名认为标志某种数据结构，或者变量。

✓ 运算符规则表面繁琐，其实井然有序，需要严格记忆；
 ✓ 特殊符号，通常担当着某些特殊功能，需要牢记。例如，\$（标志变量的“字符型属性”），*（标志“注释行”的开始，可以视为“单行注释行”的引导词），/*...*/标志多行注释的开始和结尾。☞☞

关于程序中出现的各种语汇，必须满足基本的游戏规则，否则SAS会发疯:) 这些游戏规则，就是词法。严格而言，词法指的就是语汇之间的基本规则。包括两个基本方面：一个是关键词与标志符之间的冲突调解；另一个就是标识符本身的命名规则。

准则 3.1.1 关键词和标识符不能重复。因为关键词是保留字，有优先使用权，所以该规则可以换句话说：标志符命名必须避让关键词。

准则 3.1.2 标识符的命名，须满足以下三条：

- A. 首字符：字母或者下划线（Names must start with a letter or an underscore .）；
- B. 三类组成字符：字母、数字、下划线（Names can contain only letters, numerals, or underscores. No !@#\$\$%&*, please.）；
- C. 标志符长度：≤ 32 个字符（Names must be 32 characters or fewer in length）；

需要特别补充的是字母大小写的问题，SAS处理很特别：不加区分。为了引起注意，不妨也列为“最后一条军规”，不过，此规不是约束你的手脚，恰恰相反，是怕用户们不知道自己的“自由与权利”。

准则 3.1.3 关键字和其他标志符，均不区分字母大小写，可以混合书写 (Names can contain upper- and lowercase letters)。

注意到一个问题，标识符不区分大小写造成一个问题：既然标识符可以不区分大小写，那么最终视觉上的名字是大写、小写还是混合形式呢？SAS is insensitive to case so you can use uppercase, lowercase or mixed case.whichever looks best to you. SAS

doesn't care. The data set name heightweight is the same as HEIGHTWEIGHT or HeightWeight. Likewise, the variable name BirthDate is the same as BIRTHDATE and birThDaTe. However, there is one difference for variable names. SAS remembers the case of the first occurrence of each variable name and uses that case when printing results. That is why, in this book, we use mixed case for variable names but lowercase for other SAS names.

二 句子与句法

定义 3.2 SAS语句是由关键词、SAS名、特殊字符或运算符组成并以分号结尾的字符串。句法则是连接语汇及其他语言要素的规则。

给定了一堆正确的语汇之后，如何让其连接组成承载在某种程度上具有完整意义的句子是摆在面前的一个问题，最基本的是不能犯计算机的形式错误，或者说“不犯规”或者说“不违反句法”。在SAS中，语汇无误下，组句规则有下面三条：

准则 3.1.4 一般语句通常须以关键词作为句子开始标志；运算表达式（包括“赋值运算”）以合适的运算对象或者算符开始；单行注释行也视为语句，以*号开始；数据行必须严格按照数据输入格式排列，最后以换行的分号结尾标志；在“宏”机制下，宏语句以%或&开始，其宏内注释语句的标志为单行注释%*...；。

这条规则是比较重要的，关键词开头的意义在于能让读者能很快明白语句的含义，方便记忆和使用，但是规则是灵活的，可以因时因地而变。包括了五类对象：一般语句、运算表达式、单行注释行、数据行以及宏语句，“一条总规，四个特例”。

准则 3.1.5 语汇或数据间一般应以空格间隔；运算表达式、字符属性标志\$以及句子结束标志分号可以没有空格。

这个规则显然是合理的，为了区分不同语汇，特别是标识符若没有“适当距离”，实在难分彼此，甚至难分个数，也是“一条总规，三个特例”。然而，在运算表达式中，由于区分明显，可以不加空格，有的时候为了表达紧凑甚至刻意不加空格，实在可以理解。而对于字符属性标志符*，为了表明自己是前面变量的属性标志，有时候“紧跟”主人还不够甚至“贴上”自己的主人，以显亲近也。句子标志当老大，不会与其他任何符号混淆，即使贴在任何“他者”的屁股后边，也是可以的，毕竟——“谁能不识君”呢？

准则 3.1.6 每条语句都须以分号“；”作为结束标志(Every SAS statement ends with a semicolon)。

如果说，上两条规则总规定某些合理的特例，那么只有这条规矩就是“铁规”，是没有商量的，也是判断一段语汇是否为组成句子的计算机标志。在计算机语言中，句子标志很多，例如：在早期的数据库语言Foxbase中，句子是以回车为标志，对于回车符计算机是能够感应的，但是可苦了俺们肉眼凡胎的老百姓——看不见，实在不人道也；在某些语言中，也有其他标志。分号作为句子标志，是人道的，正因为有分号保驾，SAS句子写法很多，其鉴别也单一而明显。有以下两条特别是需要注意的：

 分号是句子的标志。特别的，SAS注释有两种形式：单行注释，以分号“；”结尾，视为句子；多行注释，以/*...*/为标志，无分号标志，不视为句子。

☞ 正因为分号是句子标志，因而一个句子可以写成多行；多个句子可以写在一行。然而，从良好风格或者程序的可读性起见，最好能保证“一行一句”。

三 程序与章法

定义 3.3 SAS程序是由一些SAS语句按一定规则组成的序列，并将被依次执行的集合，通常被分成几个语句块：数据步，过程步及全程语句。

☞ 几点说明 ☞：

✓ 一个完整的SAS程序是一个序列，这些语句会依次执行(除了内循环“loop”或者控制结构)；其有别于其他语言的一个基本特点是将会形成了几个语句块：数据步，过程步和其他语句块。

✓ 数据步，可以完成数据的输入、简单的数据处理和数据输出，其基本功能为数据输入，即建立数据集。其基本标志为：首句为Data步语句，末句为Run语句。

✓ 过程步，可以完成数据的输入、复杂的数据处理和数据输出，其基本功能为复杂的数据处理。其基本标志为：首句为Proc步语句，末句为Run语句。

✓ 全程语句是特立独行的，但绝不是无用的，其作用范围广，需要慎重使用，善加使用。☞☞

There really aren't any rules about how to format your SAS program. 然而，还是有某些需要遵守的。SAS程序的规矩也是比较特别的，如果每条语句都不违规的情况下，篇章须满足以下几条规矩：

准则 3.1.7 不同的语句只能呆在特殊的语句块，要么数据步、要么过程步，要么以全程语句出现——待在“两步”之外，最后很特殊也很有SAS特色的宏语句，则几乎可以待在任何位置。具体而言，有：

A. 变量的运算表达式、数据集的运算表达式、循环和分支控制语句只能呆在数据步，因为其被Base模块支持；

B. 矩阵的运算表达式只能呆在IML过程步中(注意IML可以进行交互命令形式出现，但是已经不属于编程规矩问题了)；

C. ODS系统语句某些呆在数据步，某些可以在数据步，还有某些须在两步之外；其他一些相关语句分野明显，大体以其基本功能即可区分：与数据输入相关者在数据步，与数据的高级分析处理有关者大体在过程步，全程语句在两步之外。

D. 宏语句可以待在任何想要的位置，它决定着“子函数”或者“宏变量”的生存期。当然，比较特殊的一点要求是：宏变量的引用不能出现在单引号中(但可以出现在双引号中)

上述规则描述了SAS程序的一个重要特点，语句与语句块的关系密切，就像人不能走错人群、站错队伍，否则苦不堪言也——计算机语言，不仅如此，站错队伍就被直接枪毙了。(这条规矩不是SAS特有，例如：在c语言中，某些INCLUDE 语句就不能跑到子函数里去，否则也就被“枪毙了”；然而，但这条规矩被这么隆重强调的，却只有SAS，因为SAS承担了特别复杂繁重的分析处理任务，各条语句需要明确细致的分工，需要各安其所，各司其职。

下面的规矩描述了程序不能“言之无物”，须有基本的“块”。

准则 3.1.8 一个SAS程序至少需要一个DATA步或一个PROC步。

为了便于记忆，我们也搞出三条“军规”来，针对崇尚自由的人们，章法也来了一个“自由权”的特别规定：

准则 3.1.9 当有多个数据步与过程步同时出现时，则后一个PROC语句或DATA语句将起到前一步的RUN语句的作用，故中间的RUN语句可以省略；最后一步必须有RUN语句，否则将不能运行；

例 3.1.1 输入100位学生的身高(H)、体重(w)的数据，并引入以身高为分组变量的数据集。最后做相应的描述性统计分析。其经典程序如下：

```
* S A S 程序，主要计算学生身高、体重，并以身高为基础引入分类变量；
/* 文件名为
EXAMP11.SAS */

data wh100;
    input h w @@;
    hgroup=int((h-155)*10/(185.5-155)-0.001)+1;
    cards;
172.4 75.0 169.3 54.8 169.3 64.0 171.4 64.8 166.5 47.4 171.4 62.2
168.2 66.9 165.1 52.0 168.8 62.2 167.8 65.0 165.8 62.2 167.8 65.0
164.4 58.7 169.9 57.5 164.9 63.5 160.3 55.2 175.0 66.6 172.5 73.5
172.0 64.0 168.4 57.0 155.0 57.0 175.5 63.9 172.3 69.0 168.6 58.0
176.4 56.9 173.2 57.5 167.5 50.0 169.4 52.2 166.7 72.0 169.5 57.0
165.7 55.4 161.2 48.5 172.8 57.0 175.1 75.5 157.5 50.5 169.8 62.9
168.6 63.4 172.6 61.0 163.8 58.5 165.1 61.5 166.7 52.5 170.9 61.0
166.1 69.5 166.2 62.5 172.4 52.6 172.8 60.0 177.8 63.9 162.7 56.8
168.8 54.0 169.1 66.2 177.5 60.0 177.0 66.2 169.9 55.9 167.4 54.4
169.3 58.4 172.8 72.8 169.8 58.0 160.0 65.3 179.1 62.2 172.3 49.8
163.3 46.5 172.9 66.7 165.4 58.0 175.8 63.2 162.3 52.2 165.4 65.7
171.5 59.3 176.6 66.3 181.7 68.6 175.2 74.9 169.5 59.5 169.6 61.5
169.1 63.1 185.5 77.0 173.9 65.5 162.5 50.0 171.5 58.5 175.6 59.8
166.0 75.5 167.2 63.3 171.9 57.0 176.6 58.4 177.3 67.0 169.2 71.8
166.2 49.8 181.7 63.0 175.8 68.3 172.3 55.5 172.7 58.5 174.3 64.0
171.2 59.0 174.8 68.0 165.4 55.5 169.1 64.8 167.9 62.0 176.8 64.0
183.5 69.9 165.5 48.6 171.0 70.5 170.3 58.5
;

proc print data=wh100; run;

proc means data=wh100 mean std cv skewness kurtosis;
    var h;
run;
```

```
proc freq data=wh100;
  tables hgroup;
run;

proc chart data=wh100;
  vbar h / midpoints=158 to 186 by 4;
run;

proc chart data=wh100;
  vbar hgroup / discrete;
run;

proc univariate data=wh100 plot;
  var h;
run;
```

该程序是一个经典程序，我们现在只需要大体读懂其结构。有某些重要的编程技巧(如：连续变量的离散化)，也有一些常见的过程步及相应的使用技巧，还有一般的描述性统计学的分析传统步骤，后面还要专门剖析该程序。

下面用一个视图展现SAS的基本篇章结构，揭示“数据步”和“过程步”的整体逻辑关系：

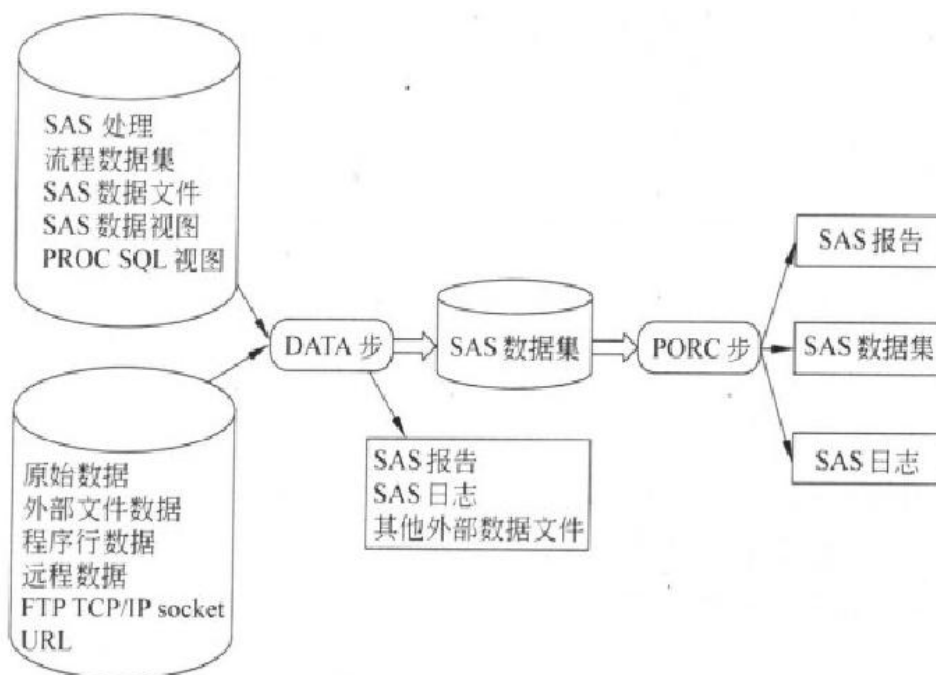


图 3.1: SAS程序结构

§3.1.3 SAS程序的生命历程

一 孕育—编辑、查错与提交

✓ 编辑与语法查错—Debugging 系统

如果只是简单的代码输入以及简单编辑，方式可谓无穷无尽，因为这只是一些符号文本而已，几乎所有的文本编辑工具都可以完成，例如：①可以直接从SAS的编辑器(Editor)编辑；②也可以从其他任何格式的文本从FILE菜单中读入，其默认的前提就是你可以从其他任何编辑器编辑好；③用剪贴板把在其他地方编辑好的拷贝过来。

然而，为了获得有风格良好、语法正确且逻辑简明的SAS代码，我们需要一定的“编程修养”以及较好的编辑工具，SAS编辑器由Debugging 系统支持，能很好的支撑具有特别风格的SAS代码编辑，包括输入、修改（复制，删除，剪切，粘贴）等等以及语法查错。

首先谈谈“编程修养”的第一个问题—风格良好或者可读性（Readable）。好的程序代码，起码应该具备“可读性”（形式），然后才讲求“语法的正确”（Correct Syntax）与“逻辑的简明”（Clear Logic）。One simple thing you can do is develop the habit of writing programs in a neat and consistent manner. Programs that are easy to read are easier to debug and will save you time in the long run. 关于“可读性”（Readability），主要有以下几条：

- 1). “一行一句” —Put only one SAS statement on a line.

SAS allows you to put as many statements on a line as you wish, which may save you some space in your program, but the saved space is rarely worth the sacrifice in readability.

- 2). “缩进式” —Use indention to show the different parts of the program.

Indent all statements within the DATA and PROC steps. This way you can tell at a glance how many DATA and PROC steps there are in a program and which statement belongs to which step. It's also helpful to further indent any statements between a DO statement and its END statement.

- 3). “多加注释” —Use comment statements generously to document your programs.

This takes some discipline but is important, especially if anyone else is likely to read or use your program. Everyone has a different programming style, and it is often impossible to figure out what someone else's program is doing and why. Comment statements take the mystery out of the program.

而对于“语法差错”以及“逻辑错误”，分别有以下工具进行测试，有些属于“编程修养”的一部分，需要作为习惯固化(如：下面第一条)

- 1). 随时查看程序的每一个部分，特别是查看数据集。

Test each part of the program. You can increase your programming efficiency tremendously by making sure each part of your program is working before moving on to write the next part. If you were building a house, you would make sure the foundation was level and square before putting up the walls.

If you are reading data from a file, use Viewtable or PROC PRINT to check the SAS data set at least once to make sure it is correct before moving on. It's a good habit to look at all the SAS data sets you create in a program at least once to make sure they

are correct. As with reading raw data files, sometimes merging and setting data sets can produce the wrong result even though there were no error messages. So when in doubt, use Viewtable or PROC PRINT.

2). 用数据集测试程序。

Test programs with small data sets. If you are reading data from a file, you can use the OBS= option in the INFILE statement to tell SAS to stop reading when it gets to that line in the file. This way you can read only the first 50 or 100 lines of data or however many it takes to get a good representation of your data.

Test with representative data. Using OBS= and FIRSTOBS= is an easy way to test your programs, but sometimes it is difficult to get a good representation of your data this way. You may need to create a small test data set by extracting representative parts of the larger data set. Or you may want to make up representative data for testing purposes. Making up data has the advantage that you can simplify the data and make sure you have every possible combination of values to test.

3). 用Debugging系统测试语法错误。

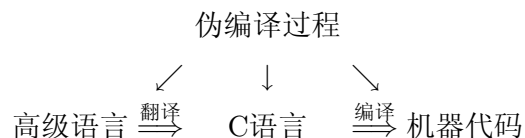
Syntax sensitive editors In the Windows operating environment the Enhanced Editor is the default editor; in other operating environments the Program Editor is the default. Both the Enhanced Editor and the Program Editor color code your program as you write it. SAS keywords appear in one color, variables in another. All text within quotation marks appears in the same color, so it is immediately obvious when you forget to close your quotation marks. Similarly, missing semicolons are much easier to discover because the colors in your program are not right. Catching errors as you type them can be a real time saver.

✓ 提交

提交就像孕妇送往医院，即将面临“分娩”。有三种提交方式：①“小人”提交；指快捷图标的“奔跑小人”②命令提交；指命令栏中输入“run”命令；③菜单提交；从run菜单中运行。三种本质一样，从一个小小的视角展现了SAS系统的功能的变化多样的执行方式。

二 分娩——伪编译过程

要真正让计算机读懂SAS语言，需要对其进行编译，将“高级语言”翻译成“机器语言”，这个过程不是直接的，大体如下：



这是一个伪编译过程，他是先把程序解释为C语言，然后再执行编译操作，因此我们也可以说：这是一个解释执行过程，并不像c等这些面向过程的语言，直接变成机器语言被计算机执行，所以SAS的这种解释执行过程被称为“伪编译过程”。其错误将在LOG窗口中详细显示。

综上所述，SAS语言可以说是面向问题的高级语言，她只需用户明白“做什么”（What to do），而不需要明白“怎么做”（How to do），这样就把我们从具体过程中解放出来，为

我们处理数据节省了宝贵的精力。对于这种语言我们应该把精力更多的放到问题的提出逻辑，或者说把我们的精力集中于去设计好数据分析的基本思路和步骤，而不要太关注执行的过程，只有这样才能尽得SAS数据分析的功能之精华与实现之愉悦。

三 生命期——“内存得失”的故事

定义 3.4 SAS程序的生命期指的是以内存占有作为开始标志到内存放弃为终止标志的过程。

程序生命的本质：占据一定的内存空间

计算机程序开始的标志很多，一般以其开始运行代码为标志，等价的说必须占有一定的内存空间，这些内存空间有两个意义，一个是存储变量和其他数据，另一个就是存储指令。当然，程序开始后可能很长一段时间也占有其他计算机资源，包括CPU进行计算，或者打印机等外设，但是唯一离不开的就是内存。

生命灭亡的标志：放弃所占有的内存空间

计算机程序结束的标志，通常认为有两个，一个是从关闭程序编辑窗口，另一个认为就是退出内存占有。很多人以前一个为标志，这是表象的，不准确的，有人关闭编辑器，但是却由于没有退出内存占有，使得某些变量仍然占有内存，其实此时SAS程序还处于生命状态；相反的，后者才是本质，如果退出了内存，即使没有关闭编辑器，一般认为程序生命已经结束，或者说处于新的“孕育期”；

有始有终的程序习惯：内存管理，节省资源

懂得了程序的生命概念，应养成好的程序习惯。特别是资源节省的概念，主要指内存资源。当要退出某一个程序时，应先关闭内存空间的占有，特别是比较大的程序，包括IML等。

§3.2 SAS表达式

§3.2.1 数据集与数据库

Before you run an analysis, before you write a report, before you do anything with your data, SAS must be able to read your data. Before SAS can analyze your data, the data must be in a special form called a SAS data set. 因此，SAS数据集是最重要的概念，它既是SAS系统操作的对象，又是数据在SAS系统中的存储形式。数据集通过数据库组织在一起。

一 数据集

定义 3.5 数据集：由若干行与若干列组成的表格。与关系数据库中的一张表类似。

● 数据集分类：

(1). 临时数据集：数据集一经建立，SAS进程未结束的时候存在；一旦退出SAS进程，则数据集随之消失。临时数据可用来保存中间结果或者练习之用

(2). 永久数据集：数据集一经建立，SAS进程未结束的时候存在；而且退出SAS进程，则数据集依然存在，下次启动SAS的时候仍然可以使用它。

根据用途选择数据的存在形式。

- 数据集和数据集文件名:

数据集名与数据集文件名:

(1). 单水平名(临时数据集, 默认为work数据库, 其形式为“数据集名”)和二水平名(永久数据集, 附带了数据库名字, 其形式为“数据库名.数据集名”)。其中数据集必须有一个名字(数据集名), 命名规则与一般标志符相同。

(2). 数据集文件是数据集的物理存储形式, 一般在硬盘上, 数据集文件名: 当存储为文件时, 其名字为: 主文件名.扩展名, 其中“主文件名=数据集名”, “扩展名=sas7bdat”。

- 数据集的结构概念:

(1). 一行(row)表示一个观察值(Observation), 它描述单一个体, 如: 一个人、一只股票、一个地区或一个线性不等式等等。按传统的SAS术语, 又常称为: 观测, 个体等, 而按关系数据库(relational database)术语又称为行, 记录。观测个数SAS没有明确限制, 取决于计算机本身的容量。

(2). 一列(column)表示一个变量(Variable), 作为变量的一种存在形式, 可以取不同的类型。按传统的SAS术语, 又常称为: 变量, 指标等, 而按关系数据库(relational database)术语又称为列, 字段, 属性。Sas 9.1最多允许32767个变量。每个数据集总有一个ID列, 用以标志记录顺序。

(3). 观测值(value)

- 数据集(文件)中的基本信息: 文件中蕴含文档描述信息以及数据信息两类基本信息, 可以随时调用, 因而可以说是“自我存档”的(Self-documenting):

(1). 描述性信息: 数据集基本信息包括数据集名字、创建日期以及SAS版本; 变量信息包括变量名、类型、长度以及在数据集中的位置。

(2). 数据信息: 每一条记录。

二 数据库

定义 3.6 数据库: 不同于通常所说的数据库, 在SAS系统中, 数据集是通过数据库组织在一起的。也就是说, 数据库其实是比数据集高一级的目录。

- 逻辑数据库与物理路径: SAS逻辑库就相当于给一个人起的名字, 是一个逻辑标示, 真正对应的是指向的物理文件路径。当然, work库只有逻辑库而没有物理路径。

- 数据库分类: 预定义数据库和自定义数据库

(1). WORK: 用于存放临时数据集

(2). SASUSER: 用于存放用户自己的永久数据集

(3). SASHELP: 用于存放系统的帮助, 例子文件, 也是永久数据集

(4). 自定义数据库: LIBNAME 语句

libname 自定义数据库名 ‘硬\盘\路\径(文件目录)’

- 关系数据库的基本理论: 关系与关系运算。

§3.2.2 常量

一 常量的定义

定义 3.7 SAS常数用来表示在某一过程固定的值，它或者是一个数字，或者是用引号引起来的字符串、或者是其它特殊记号(主要是日期时间)。

- 常数的类型主要包含其内在的运算特性(没有空间概念，因为其不可变)。SAS使用的常数有五种类型：数值常数、字符常数、日期、时间和日期时间常数、十六进制数值常数、十六进制字符常数。

二 数值常量

数值常数：就是出现在SAS语句里的数字。很多数值常数完全像通常的数据值一样书写。数值常数可以包括小数点，负号和E记号。

例 3.2.1 数值类型常数主要包括整数(正整数，负整数，零)，小数(因为位数所限，指有理数)，E表示法

整数：1, -5, 0

小数：0.1 , 1.23,

E表示法：用E表示法时，如2E4，它表示20000，如1.2E3, 0.5E 5

数值常数缺失时用小数点"."表示。

三 字符常量

字符常数是单引号括起来的1到200个字符组成的。如语句：if name='tom' then do;中的'tom'是一个字符常数。

在下列两种情形应该用双引号：

(1). 如果字符常数含有引号，此时应用双引号括起来，如name="'tom' s"。字符常数缺失时用空格加引号(")表示。

(2). 或者出现宏引用情形，也应该用双引号

如果字符常数含有引号，则用两个连续的单引号来处理。例如，字符值为TOM'S时，输入：name='TOM'S'

缺失的字符常数值为空字符表示为"（两个连续的单引号）。

四 日期、时间和日期时间常量

为了把日期、时间或日期时间值表示为常数，在输入格式或输出格式中使用相同记法：DATE.，TIME.和DATETIME.。格式值用单引号括起来，并跟随一个D（日期），T（时间）或DT（日期时间）。

例 3.2.2 三种类型，注意其顺序，日期型：日月年；时间型：时分秒。时分秒之间用冒号分开，日期时间之间用冒号分开

日期型：'1JAN1998'D, '01JAN98'D

时间型：'9:25'T ,

日期时间型：'18JAN98:9:25:20'DT

§3.2.3 变量

一 变量的定义

定义 3.8 SAS变量用来表示在某一个过程可以变化的量。其最重要的特性是变量类型，描述了变量所取的数据值的集合及其上的运算特性(等价于数学上的“空间”概念)。

- 变量的基本属性包括两类:
 - (1). 形式属性: 变量名, 标签(可以默认, 视为变量的“别名”, “小名”)
 - (2). 内容属性: 变量类型, 存储长度(可以默认, 进一步决定取值范围大小)
 - (3). 输入输出格式属性: 输入格式、输出格式
- 变量类型(有时候结合“存储长度”), 描述了变量所取的数据值的集合及其上的运算特性(等价于数学上的“空间”概念)。SAS变量分为两类, 数值变量(Num)与字符变量(Char)。

二 数值变量

只能取数值为值。可以有正负号及小数点(+、-、.), 但不能有逗号(,), 数值型是SAS的默认变量类型, 默认长度为8位。

例 3.2.3 数值类型变量的取值主要包括整数(正整数, 负整数, 零), 小数(因为位数所限, 指有理数), E表示法。即上述的数值型常数均可以作为取值进入数值变量, 但是会由于存储长度(或者默认长度)而被“截尾”, 或者“0加长”

三 字符变量

可以取字符、字母、特殊字符以及数字为值。在DATA步中某些SAS语句所使用的变量名后跟一个美元符号(\$), 即表明该变量是字符型变量, 如name\$ 或name \$。如果变量后不加\$符号, SAS将该变量认为数值型。因此, 对字符型变量应加\$符号予以说明。

在SAS中一个字符变量的取值可以有1-200个字符长, 默认长度为8个字符长

§3.2.4 SAS函数

一 SAS函数的定义

定义 3.9 SAS函数是SAS系统中编好的子程序, 它对若干个(0个或者多个)参数进行计算后返回一个结果值。

- SAS函数的基本要素:
 - (1). 函数名: SAS函数都有一个关键词名字。
 - (2). SAS函数的参数和结果:
- SAS函数的引用:
 - (1). 为了调用一个函数, 先写出函数名, 接着是空括号或括在括号中的若干个要进行计算的参数, 其一般形式如下:

函数名 () 或 函数名 (参数1, 参数2, ...)

参数之间应该用逗号隔开，

(2). 当参数多于一个时，也可写成如下形式之一：

函数名 (OF变量1 ... 变量N) 或函数名 (OF变量1 -变量N)

- SAS函数作为变量参与运算(因为有返回结果):

SAS函数作为表达式或表达式的一部分用于DATA步的编程语句中及一些统计过程中。如作为一个赋值语句：TOTAL=SUM (X1, X2, X3)；作为一个表达式的一部分：

```
IF SUM(CASH,DATA)<1000 THEN DOLL=INT(CASH);
```

- 熟悉运用SAS函数，适当条件下能简化SAS程序,有时甚至必须使用SAS函数来进行计算。如以下程序：

SAS源码 3.2: 函数简化程序

```
1 total=x1+x2+x3+x4+x5+x6+x7+x8+x9+x10;
2 if total>y then z=total;
3 else
4 z=y;
```

可以简化为：

```
z=max (y,sum (of x1-x10) )
```

；又如，计算定积分：

$$\int_{-\infty}^1 \exp \left\{ -\frac{(x-4)^2}{8} \right\} dx$$

的值时就必须使用正态概率函数。

例 3.2.4 SUM (OF X1-X50 Y1-Y50) , SUM (OF X Y Z) , SUM (X1, X2, Y1, Y2)
这里SUM为求和函数。

二 SAS函数的参数和结果

1. SAS函数的参数

- 参数的类型：参数可以是简单的变量名、常数或者表达式。这个表达式还可以包含其它的函数。例如：

```
max(cash,credit)
sqrt(2500)
min(sum(of x1-x10),y)
```

- 参数的个数：有些函数不需要参数(如DATE()),有些只需要一个参数，有些函数是对几个参数作分析处理。如果函数的参数是表达式，则先计算作为参数的表达式例如：LOG(x+y)这个函数，是先计算x+y，然后调用对数函数LOG来计算x+y的对数

- 参数的表示法：一般地，当函数有几个参数时，它们之间必须用逗号(,)分隔开。

2. 函数的结果

函数的取值通常取决于参数的类型，当参数是字符型变量时，函数的取值为字符；当参数为数值型变量时，函数的取值为数值

三 函数的分类

1. 函数分类概述：

SAS函数非常丰富，为了方便记忆，大体可分为以下八大门类

数值运算类：算术函数、数学函数、三角函数；

字符运算类：字符函数、字符串匹配函数

日期时间运算类：日期时间函数

逻辑运算类：逻辑函数

数组运算类：数组函数

概率统计类：概率与密度函数、分位数函数、随机数函数、样本统计函数等

金融货币类：金融计算函数，货币转换函数

其他函数：截取函数、地区以及邮政编码函数、变量信息函数、目录函数

2. 数值运算类：

ABS (X)：绝对值函数；EXP (X)：指数函数；LOG (X)：以e为底的对数函数；LOG10 (X)：以10为底的对数函数；SIGN (X)：符号函数；INT (X)：取整函数；MAX (X,Y,...)：最大值函数；SQRT (X)：平方根函数；SIN (X)：正弦函数；COS (X)：余弦函数；TAN (X)：正切函数；ATAN (X)：反正切函数。

3. 日期时间类

year(date):从变量中抽取”年份”值

month(date):从变量中抽取”月份”值

day(date):返回变量的天数

mdy(月, 日, 年):将变量转换成日期值形式

4. 财政金融类

COMPOUND：计算复利系数；DEPSL：用直线折旧法计算资产的折旧额；DEPSYD：用年限总和法计算资产的折旧额；DEPDB：用余额递减折旧法计算资产的折旧额；IRR：计算用百分数表示的内部收益率；NPV：计算用百分数表示的净现值；SAVING：计算定期储蓄的本金和利息。

5. 概率与密度函数

PROBNORM (x)：标准正态分布函数，该函数计算服从标准正态分布的随机变量U小于给定x的概率；

PROBCHI (x, df, nc)： χ^2 分布的分布函数。该函数计算服从自由度为df，非中心参数为nc的 χ^2 分布的随机变量小于给定x的概率。如果nc没有规定或取为0，那么被计算的就是中心 χ^2 分布。注意，自由度df允许是非整数。

PROBT (x, df, nc) : 该函数计算服从自由度为df, 非中心参数为nc的T分布的随机变量小于给定x的概率。如果nc没有规定或取为0, 那么被计算的就是中心T分布。注意: 自由度df允许是非整数。

PROBF (x, ndf, ddf, nc) : F分布的分布函数, 该函数计算服从分子自由度为ndf, 分母自由度为ddf, 非中心参数为nc的F分布的随机变量小于给定值x的概率。如果nc没有规定或取为0, 那么被计算的就是中心F分布。注意: 自由度允许是非整数。

PROBBNML (p, n, m) : 其中, $0 \leq p \leq 1, n \geq 1, m \geq 0$, 这个函数给出参数为p和n的二项分布随机变量小于等于m的概率。

6. 随机数函数

均匀分布: UNIFORM (seed) 或RANUNI (seed), 这里seed必须是常数, 前者或是0或是5位、6位、7位的奇数; 后者是模小于 $2^{31} - 1$ 的任何数值常数。

标准正态分布: NORMAL (seed) 或RANNOR (seed), 这里seed必须是常数, 前者或是0或是5位、6位、7位的奇数; 后者是模小于的任何数值常数。

注意, 对于均值为M, 标准差为S的正态分布随机变量Y, 可由标准正态分布的线性函数得到:

$$Y = M + S * \text{NORMAL}(\text{seed})$$

例 3.2.5 试用产生标准正态分布的随机函数normal (seed) 产生均值为170, 方差为30正态随机数10个。

解: 根据题目要求, 编程如下:

```
data a (drop=i);
do i=1 to 100 by 1;
z=170+sqrt(5)*normal(0);
output;
end;
run;

proc print data=a;
run;
```

程序说明: 程序利用循环语句和赋值语句创建了一个名为a的SAS数据集, 注意这里的数据步没有INPUT语句。循环变量i从1到10表示循环10次, 而赋值语句则利用标准正态随机函数normal (0) 产生一个均值为170, 方差为30的正态分布的随机数, 从而整个循环组产生10个这样的随机数。程序所产生的10个随机数字如下: 注意, 每次产生的随

表 3.1: 由正态随机函数产生的10个随机数

167.151	164.028	176.099	176.227	179.322
174.731	168.514	168.983	173.416	172.156

机数可能不同。

7. 样本统计函数

SUM (X1,X2,...) 或SUM (OF X1 X2...): 求和函数;

MEAN (X1,X2,...) 或MEAN (OF X1 X2...): 均值函数;

MAX (X,Y,...) 或MAX (OF X1 X2...): 最大值函数;

MIN (X,Y,...) , 或MIN (OF X1 X2...): 最小值函数;

STD (X1,X2,...) 或STD (OF X1 X2...): 标准差函数

STDERR (X1,X2,...) 或STDERR (OF X1 X2...): 标准误差函数, 标准误差= s/\sqrt{n}

CV (X1,X2,...) 或CV (OF X1 X2...): 变异系数, 公式为 $CV=s/\bar{x} \times 100\%$

SKEWNESS (X1,X2,...) 或SKEWNESS (OF X1 X2...): 变量X1,X2,...等的偏度, 计算公式为:

KURTOSIS (X1,X2,...) 或KURTOSIS (OF X1 X2...): 变量X1,X2,...等的峰度, 计算公式为:

例 3.2.6 下表 3.2 是我国内地各省、自治区及直辖市2007年度城镇居民人均可支配收入资料。试根据所给出的数据计算我国内地各省、自治区及直辖市人均可支配收入的平均值、标准差、偏度和峰度。

表 3.2: 2007年度各地区城镇居民人均可支配收入资料 单位: 元

全国	北京	天津	河北	山西	内蒙古	辽宁	吉林
13785.81	21988.71	16357.35	11690.47	11564.95	12377.84	12300.39	11285.52
黑龙江	上海	江苏	浙江	安徽	福建	江西	山东
10245.28	23622.73	16378.01	20573.82	11473.58	15506.05	11451.69	14264.7
河南	湖北	湖南	广东	广西	海南	重庆	四川
11477.05	11485.8	12293.54	17699.3	12200.44	10996.87	12590.78	11098.28
贵州	云南	西藏	陕西	甘肃	青海	宁夏	新疆
10678.4	11496.11	11130.93	10763.34	10012.34	10276.06	10859.33	10313.44

资料来源: 《中国统计年鉴》2008年

解: 根据所给出的资料, 程序编辑如下:

```
data a (drop=income1-income31);
input income1-income31@@;
mean=mean (of income1-income31);
std=std (of income1-income31);
skewness=skewness (of income1-income31);
kurtosis=kurtosis (of income1-income31);
cards;
13785.81 21988.71 16357.35 11690.47 11564.95 12377.84 12300.39
11285.5 10245.28 23622.73 16378.01 20573.82 11473.58 15506.05
```



```

11451.69    14264.7    11477.05    11485.80    12293.54    17699.30    12200.44
10996.87    12590.78    11098.28    10678.40    11496.11    11130.93    10763.34
10012.34    10276.06    10859.33    10313.4
;
proc print;run;

```

程序运行结果列表如下：从输出结果可以看出，2007年全国人均可支配收入

平均值	标准差	偏度	峰度
13223.40	3527.26	1.74988	2.37162

为13223.40 元，偏度为1.74988表明，收入分布略向右偏斜。

§3.2.5 运算符与运算类型

一 运算符概述

SAS操作符是一些符号，用它们可以作比较、算术运算或逻辑运算，它有前缀和中缀之分。前缀操作符用在数值、变量或函数的前面，主要有+、-、NOT。而中缀操作符则是用在两个运算对象的中间，主要有算术操作符、比较操作符、逻辑操作符、其他操作符四类

二 算术操作符与算术运算

算术操作符表示执行一种算术运算。常用算术操作符、含义及举例见表 3.3

表 3.3: SAS算术操作符举例

操作符		运算对象和结果		举例
操作符	符号含义	运算对象	运算结果	
+	加法	数值型	数值型	Sum=x+y
-	减法	数值型	数值型	Diff=x-y
*	乘法	数值型	数值型	Mult=x*y
/	除法	数值型	数值型	Divide=x/y
**	幂运算	数值型	数值型	Raise=x**5
	字符串并接	数值型	数值型	Str=str1——str2

三 比较操作符与比较运算

比较操作符用来建立两个量之间的一种关系，并要求SAS确定这种关系是成立或不成立。如果成立，输出的结果为1；如果不成立，结果为0。常用比较操作符、含义及举例见表 3.4

表 3.4: SAS比较操作符举例

操作符		运算对象和结果		举例
操作符	符号	运算对象	运算结果	
LT	<	数值型或者字符型	逻辑型	if x1<5;
GT	>	数值型或者字符型	逻辑型	if price<10;
EQ	=	数值型或者字符型	逻辑型	if dest='lon';
LE	<=	数值型或者字符型	逻辑型	if output<=1000;
GE	>=	数值型或者字符型	逻辑型	if cost<=1000;
NE	≠	数值型或者字符型	逻辑型	if x1≠5;
IN		数值型或者字符型	逻辑型	if dest in ('lon', 'par') ;

四 逻辑操作符与逻辑运算

逻辑操作符也称为布尔算符，在表达式里通常用来连接一系列比较式，常与IF语句结合使用。常用逻辑操作符、含义及举例见表 3.5

表 3.5: SAS逻辑操作符举例

操作符		运算对象和结果		举例
操作符	符号	运算对象	运算结果	
OR		逻辑型	逻辑型	if x<3 or x<4;
AND	&	逻辑型	逻辑型	if x<3 and x<8;
NOT		逻辑型	逻辑型	if x^3 ;

五 运算次序

以上SAS操作符与其他高级语言的操作符在符号、含义及运算顺序上基本是一致的，具体而言有以下三个基本准则：

准则 3.2.1 （括号）：在括号里的表达式先计算；

准则 3.2.2 （优先级）：较高优先级的运算先被执行

下面是各运算符的优先等级(排在前面的优先)：

1级 **、(not)

2级 *、/

3级 +、-、

4级 <, <=, =, >, >=, >, <

5级 &(and)

6级 |(or)

准则 3.2.3 （左结合性）：对于相同优先级的算符，左边的运算先做。但有两个例外：

(1). 对第一级，右边的先做；

$2**3**2 \quad \leftarrow \rightarrow \quad 2**(3**2)$

(2). 当两个比较算符围着一个量时，则等价于一个and运算。例如：

$12 < \text{age} < 20$

$\leftarrow \rightarrow \quad 12 < \text{age} \text{ and } \text{age} < 20$

§3.2.6 表达式

SAS表达式是由一系列操作符与运算对象连接而成的，它被执行后产生一个运算结果。运算对象可以是常数、变量和函数，操作符可以是算术操作符、比较操作符和逻辑操作符。不含或仅含有一个操作符的表达式称为简单表达式，含有二个或二个以上操作符的表达式称为复合表达式。下列式子都是表达式。

$X+5, \quad 10, \quad \text{SIN}(X), \quad (X1+X2)/2,$

$X+10 \text{ EXP}(X1 \text{ } X2), \quad \text{SCORE} > 90, \quad A=B=C$

注意，单独的常数、变量也是表达式的一种形式。

§3.3 数据集与矩阵

§3.3.1 数据集及其查询与运算（SQL过程）


§3.3.2 矩阵运算与IML过程

一 IML语言概述

二 矩阵及其表达式


1. 矩阵定义

 矩阵名：与变量名等一般的标识符的命名规则相同。

 矩阵的定义与赋值同时产生。赋值方法分为两种：

a. 原始值输入；

b. 表达式输入；

 “原始值”矩阵定义规则：

a. 大括号标志；

b. 同行内数据以空格标志；

c. 不同行数据以逗号“，”标志。

d. 同行内重复元表达式：“[n] x”，即为 n 个 x 在同一行。

 “表达式”矩阵定义规则：记住常见的表达式。

例 3.3.1

2. 矩阵线性运算:

矩阵加法: 矩阵对应元素相加, 其前提是同型。

矩阵减法: 矩阵对应元素相减, 其前提是同型。

矩阵数乘: 矩阵每个元素分别相乘。

相反数运算: 矩阵每个元素乘以-1

3. 矩阵乘法

矩阵乘法: $a * b$ (前提是 a 的列数等于 b 的行数)

矩阵乘方: $a ** n$ (前提是 a 是方阵)

4. Kronecker乘积

相乘获得新矩阵, 其行数等于两矩阵行数乘积, 其列数等于两矩阵列数乘积, 不可交换。

$$a @ b \quad (3.1)$$

5. 转置运算

a'

6. 其他运算

对应元素相乘: $a \# b$ (前提是同型)

对应元素乘方: $a \# \# n$ (无前提)

对应元素相除: a / b (前提是同型)

比较运算(<, >, =, =<, >=)和逻辑运算(|, & (前提是同型)

取最大值(<>)与取最小值(><)(前提是同型)

表运算: 水平连接(||, 同行数), 垂直连接(//, 同列数)

三 矩阵模块

四 矩阵与数据集的交互

§3.4 Data步机理与常见语句

§3.4.1 Data步语言概述与内循环(Built-in Loop)

一 Data步及其语言概述

DATA steps start with the DATA statement, which starts, not surprisingly, with the word DATA. This keyword is followed by a name that you make up for a SAS data set. In addition to reading data from external, raw data files, DATA steps can include DO loops, IF-THEN/ELSE logic, and a large assortment of numeric and character functions. DATA steps can also combine data sets in just about any way you want, including concatenation and match-merge.

data步语言组成一套非常完整的程序设计语言体系。一方面, 她有一般编程语言所必备的语言架构, 包括表达式以及其他可执行语句; 表操作语句; 数组语句; 控制语句等。表达式问题可以说是data步非常中心的问题, 但是不在此多述, 只是因为它更多的牵涉到一般的数据集问题, 因而放在一般情形; 另一方面, 她有别的语言所没有的

特有的“内循环”，这是Data的重要机理，This basic concept is rarely stated explicitly. Consequently, new users often grow into old users before they figure this out on their own.

其基本功能包括：

- 1). 由原始数据创建SAS变量和数据集(SAS数据文件或者SAS数据视窗);
- 2). 读取外部数据文件，创建SAS数据集;
- 3). 数据管理和操作;
- 4). 数据的基本统计分析;
- 5). 数据呈现：产生报告，或者将文件存储到硬盘或磁带上;

将数据步的整个生命期可以分解为两个基本部分(不包括前期文本编辑、修改)：一是编译过程；二是运行过程，其整体的运行流程如下：

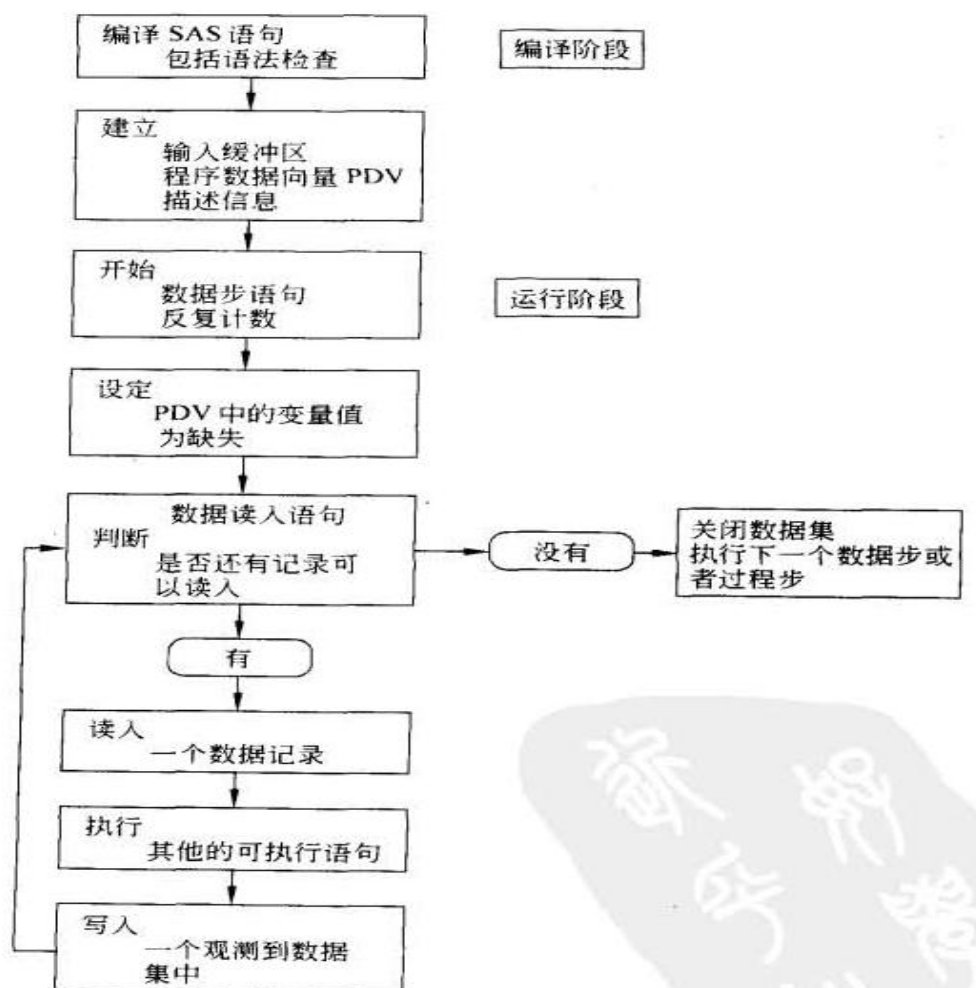


图 3.2: DATA 步流程示意图

需要特别强调以下几点：

注 3.4.1 编译阶段(The Compilation Phase): 提交数据步程序后，需要完成以下步骤

- 1). SAS首先检查这段程序的语法是否正确，

2). 然后进行编译, 自动编译为机器代码(有时候也称“伪编译”, 因为是先转化为其他语言, 然后变为机器代码)

3). 整个阶段, SAS识别每个新变量的长度、类型, 并决定每个变量是否需要类型转换。最终将创建三个项目:

表 3.6: 编译过程中SAS创立的三个项目

项目	说明
输入缓冲	内存中的一个逻辑区域。输入缓冲是一个存储区域, 只有在input语句读入外部数据源时才使用。读入SAS数据集时, SAS将直接读入PDV程序数据向量
程序数据向量 (PDV)	内存中的一个逻辑区域。SAS创建数据集时, 该区域每次读取一个观测, 包括两部分: 基础数据集变量值和赋值语句下的新变量, 前者依赖于目标数据集指针 <code>_N_</code> , 后者则首先赋予缺失值, 再获得具体值时, 覆盖缺失值; 运行程序时, 先从缓冲区域中读入建立PDV, 在把数据值分配给PDV中合适的变量, 然后将这些值作为一个观测写入数据集。除了数据集原有的变量和新创建的变量外, PDV还包含两个自动变量 <code>_N_</code> 和 <code>_ERROR_</code> 。前者记录DATA步运行的次数, 后者标记程序是否发生错误。两者都不包含在新创建的数据集中。
描述信息	描述SAS创建和保存数据集的相关信息: 数据集的属性以及变量的属性等。

资料来源: 朱世武《SAS编程技术》

注 3.4.2 运行阶段(The Running Phase), 默认情况下, 每增加一个观测都要执行一次DATA步, 简单DATA步流程的基本步骤包括:

- 1). 以DATA步开始, 每个DATA步都是一个重复开始, 这时自动变量 `_N_` 加1;
- 2). PDV的新变量的数据获取: PDV中新创建的程序变量设定为缺失状态;
- 3). PDV的基础数据集观测获取: 将基础数据集中的数据读入到输入内存中, 或者直接从SAS数据集中读取一个观测写入到PDV中。可以用INPUT,MERGE,SET,MODIFY或者UPDATE语句来读入一个记录行;
- 4). 对当前的数据记录执行程序语句;
- 5). 语句的最后自动产生一个输出、返回或者重置。这时SAS将一个观测写入到数据集中, 系统自动回到数据步开始, PDV 中由INPUT 语句创建的变量或者赋值语句创建的变量重新被设定为缺失。但是用INPUT,MERGE,SET,MODIFY或者UPDATE语句来读入的变量不会被重置为缺失。
- 6). 进入下一次重复过程: 读入下一条记录, 执行语句。
- 7). 遇到数据集结尾, 或者原始数据结尾时, 结束DATA步。

二 Data步机理—Built-in Loop, 行指针与列指针

1. 内循环

DATA steps read and modify data, and they do it in a way that is flexible, giving you lots of control over what happens to your data. However, DATA steps also have an underlying structure, an implicit, built-in loop.

定义 3.10 内循环：DATA steps also have an underlying structure, an implicit, built-in loop. You don't tell SAS to execute this loop: SAS does it automatically. Memorize this:

DATA steps execute line by line and observation by observation.

- 内循环示意图：

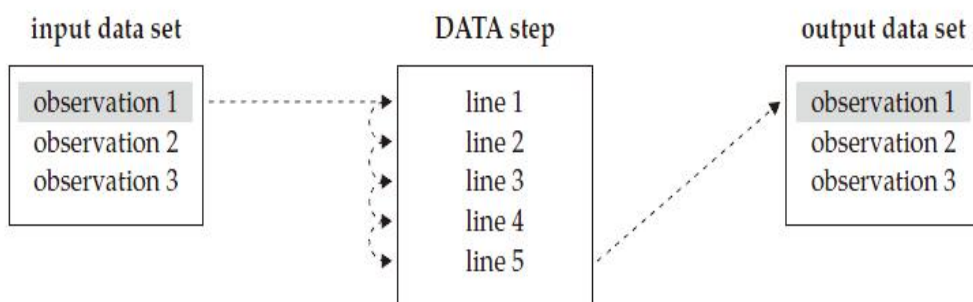


图 3.3: built in loop

- 类比：排队(记录排队)+办事手续(处理过程)

2. 行指针

3. 列指针

§3.4.2 Data步标志语句

— data语句

该语句标志DATA步的开始，同时承担一定的功能。

一般格式：**DATA** [data-set-name-1[data-set-options-1]]... ;

DATA语句表示一个数据步的开始，并给出正被创建的SAS数据集的名字。其中：

data-set-name: 表示在这个DATA步中将要建立的数据集名字，可以是一个或多个。

Data-set-options: 告诉系统关于正被创建的这个数据集的更多信息。这些选项用括号括起来并跟在相应的数据集名字的后面。例如：

```
data new(drop=y);           /*列出不包含在数据集中的变量y*/ data
new(keep=x1 x2);          /*列出包含在数据集中的变量x1,x2*/ data
new(label='health club data'); /*给数据集一个附加标签*/ data
new(rename=(x1=u x2=v)); /*把new中的变量x1,x2更名为u,v*/
```

二 run语句

Data步结束标志: **run;**

§3.4.3 Data步可执行语句（一）——基础数据操作（行列运算）

SAS系统提供了一些语句对变量或观测进行加工处理。在DATA 步建立一个SAS数据集之后，可以用SAS语句修改数据，选择观测子集，对数据加工处理等。

一 赋值语句和累加语句

1. 赋值语句

一般格式: **variable = expression;**

赋值语句将表达式计算的结果赋给变量（variable）。表达式中可以包含SAS操作符，用来执行基本的运算，还可以包含SAS函数，用来进行一些数学运算、计算统计量、处理SAS数据。需要注意的是：表达式中的变量必须已被赋值，否则作为缺省值处理。

例 3.4.1 $X=Y+Z*3;$

$X=\text{SUM}(Y, Z);$

$Y=1-\text{EXP}(3.1415*180);$

$C=8*(X<Y)+(X\geq Y);$

2. Retain语句

When reading raw data, SAS sets the values of all variables equal to missing at the start of each iteration of the DATA step. These values may be changed by INPUT or assignment statements, but they are set back to missing again when SAS returns to the top of the DATA step to process the next observation. RETAIN and sum statements change this. If a variable appears in a RETAIN statement, then its value will be retained from one iteration of the DATA step to the next. A sum statement also retains values from the previous iteration of the DATA step, but then adds to it the value of an expression.

Use the RETAIN statement when you want SAS to preserve a variable's value from the previous iteration of the DATA step. The RETAIN statement can appear anywhere in the DATA step and has the following form, where all variables to be retained are listed after the RETAIN keyword:

RETAIN variable-list;

You can also specify an initial value, instead of missing, for the variables. All variables listed before an initial value will start the first iteration of the DATA step with that value:

RETAIN variable-list initial-value;

3. 累加语句(Sum Statement)

一般格式: **variable+ expression;**

累加语句将表达式的计算结果加到累加变量上，作为累加变量新的观测值送到数据集中。这里的Variable: 定义累加变量的名字，它必须是数值型的，且第一个观测被读入

前它自动地被赋值为0。表达式 (expression) 中同样可以包含SAS操作符, 用来执行基本的运算, 还可以包含SAS函数, 用来进行一些数学运算、计算统计量、处理SAS数据。当这个表达式的计算结果为缺省值时, 它被作为0处理。A sum statement also retains values from the previous iteration of the DATA step, but you use it for the special cases where you simply want to cumulatively add the value of an expression to a variable. A sum statement, like an assignment statement, contains no keywords. It has the following form:

例 3.4.2 下列语句都是累加语句:

```
n+1; x1+x2; x1+(x2); sumx+x*x; nx+x $\hat{=}$ ;
```

No, there is no typo here and no equal sign either. This statement adds the value of the expression to the variable while retaining the variable's value from one iteration of the DATA step to the next. The variable must be numeric and has the initial value of zero. This statement can be re-written using the RETAIN statement and SUM function as follows:

```
RETAIN variable 0;
variable = SUM(variable, expression);
```

例 3.4.3 下列是2002年某班部分学生语文、数学和英语考试成绩,试利用累加语句对总分超过240的学生进行计数。考试成绩如下:

82, 78, 69, 90, 78, 89, 79, 86, 98, 76, 56, 80, 72, 76, 81, 69, 78, 91, 92, 71, 85

解: 根据题目要求, 编写程序如下:

```
data class; input chinese maths english @@;
total=chinese+maths+english; if total>=240 then n+1;else delete;
cards; 82 78 69 90 78 89 79 86 98 76 56 80 72 76 81 69 78 91 92 71
85 ;
```

程序说明: data步首先创建了一个名为class的SAS数据集。第三个语句为赋值语句, 它把语文、数学和英语成绩的总分加起来赋给变量total。第四句为条件语句, 这将在以下介绍, 其中if total>=240 then n+1语句中的n+1为累加语句, 这里n为累加变量名, 1为表达式。该语句表示当总分total>=240时, 变量n加1。程序运行结果, 请读者自行给出。

二 input语句、Cards语句与DATALINE语句

1. input语句

一般格式: INPUT语句有:表输入(自由输入)、列输入、格式化输入等方式。我们只介绍表输入方式

```
INPUT variable-name-list [$] [@@];
```

功能：描述一个输入记录中数值的排列情况，同时给相应的变量赋予输入值。INPUT语句只能用来读入跟在CARDS语句之后的数值或存放在外部文件中的数据。如果数据已经在一个SAS数据集中，则要使用SET、MERGE、UPDATE或MODIFY语句。\$用来指出它前面的变量是字符型的。如果省略\$，则默认前面的变量为数值变量。至于[@@]这个记号，我们放在后面去讨论表输入方式的条件：

- (1)输入的数据值之间至少有一个空格；
- (2)使用圆点“.”而不是空格作为缺失值；
- (3)input语句中变量的顺序和它们的数据值在数据行中的顺序一致；
- (4)字符型数据值的最大宽度为8个字节，但可以使用LENGTH、INFORMAT等语句重新定义字符串的宽度。

表输入方式的特点：表输入方式是使用最方便的输入方式，用户只需在INPUT语句中列出要被赋值的变量名即可，而不必知道数值占哪几列。正是因为如此，表输入方式又称为自由输入方式。行保持符“@@”的使用

2. CARDS 和CARDS4 语句

一般格式：

```
CARDS;
数据行
;
```

功能：如果SAS系统使用的数据是在作业流中输入的，在数据行之前用CARDS语句，用以告诉系统下面跟着的是数据行。

几点注意：

- (1)CARDS语句是DATA步的最后一个语句，它必须与INPUT语句配合使用，并且在一个数据步中最多只能用一个CARDS语句；
- (2)在数据行开始后，一旦出现分号，SAS系统确认是数据行结束；
- (3)如果在数据行中含有分号作为输入数据，则将CARDS语句改为CARDS4语句

3. DATALINE语句

4. INFILE 语句

三 output语句和delete语句

1. output语句

OUTPUT语句规定SAS系统输出当前的观测到指定的SAS数据集中。

一般格式：**output <数据集名1><数据集名N>;**

output后的括号中的数据集名是可选项。当没有该选项时，SAS系统把当前这个观测输出到DATA语句命名的所有数据集上，并返回到DATA步开始接着处理下一个观测。当有该选项时，SAS系统把当前这个观测输出到output语句规定的所有数据集上，并返回到DATA步开始接着处理下一个观测。这里数据集名可以多于一个，但必须在DATA语句中已被命名。下列例3.4、例3.5两段程序给出了OUTPUT语句常用的两种使用方法。

例 3.4.4 程序如下：

```
data class;
input name$ chinese maths english @@;
score=chinese;output;
score=maths;output;
score=english;output;
cards;
a 82 78 69 b 90 78 89 c 79 86 98
;
```

分析： 数据集class的输出结果如下：

表 3.7: 数据集class的数据结构

OBS	NAME	CHINESE	MATHS	ENGLISH	SCORE
1	a	82	78	69	82
2	a	82	78	69	78
3	a	82	78	69	69
4	b	90	78	89	90
5	b	90	78	89	78
6	b	90	78	89	89
7	c	79	86	98	79
8	c	79	86	98	86
9	c	79	86	98	98

由输出结果可以看出，三个OUTPUT语句把原始数据中的一个观测变成了三个观测。

例 3.4.5 程序如下：

```
data class1 class2;
input name$ chinese maths english @@;
total=chinese+maths+english;
if total>=240 then output class1;
else output class2;
cards;
a 82 78 69 b 90 78 89 c 79 86 98
;
proc print data=class1;
title '数据集class1';
proc print data=class2;
title '数据集class2';
run;
```

分析： 数据集class1、class2的输出结果如下：

表 3.8: 数据集class1、class2的数据结构

数据集class1					
OBS	NAME	CHINESE	MATHS	ENGLISH	SCORE
1	b	90	78	89	257
2	c	79	86	98	263

数据集class2					
OBS	NAME	CHINESE	MATHS	ENGLISH	SCORE
1	a	82	78	69	229

由输出结果可以看出，第一个OUTPUT语句把满足total \geq 240的观测输出到数据集class1中，把不满足total \geq 240即total $<$ 240 的观测输出到数据集Class2中。

2. delete语句

DELETE语句要求SAS系统停止处理当前的观测，即当前的这个观测不被输出到正创建的SAS数据集中，而且返回到DATA步的开始处理其他观测。

一般格式： **DELETE;**

例 3.4.6 下程序如下：

```
data class3 ;
input sex$ chinese maths english @@;
if sex='m' then delete;
cards;
m      82      78      69      f      90      78      89      m      79      86      98
;
```

分析： 程序输出结果如下：

表 3.9: 由if条件语句创建的数据集class3的数据结构

OBS	SEX	CHINESE	MATHS	ENGLISH
1	f	90	78	89

该程序表明：当SAS系统读入第一个观测时，由于sex='m'，所以第一个观测没有被输出到class3数据集中。当SAS系统返回到data步的开始继续处理第二个观测时，由于sex='f'，所以第二个观测被读入到class3数据集中。同理第三个观测不包含在class3数据集中。

3. OUTPUT和DELETE的混用

当两者同时出现时，并且出现矛盾信息时，以第一个出现的为准。

四 keep语句和drop语句

用户在创建SAS数据集时，有时需要数据集保留部分变量或删除一部分变量，这时就要使用SAS系统提供的KEEP语句和DROP语句。

1. keep语句

在DATA步中，KEEP语句用来规定哪些变量被包含在正被创建的SAS数据集中。该语句适用于正被创建的所有SAS数据集，而且出现在任何地方其作用也相同。当在一个DATA步中同时创建两个或两个以上不同的SAS数据集时，为了区分每个数据集所要包含的特殊变量，需要在该数据集后面使用KEEP= variables选项，它与KEEP语句的作用是相同的。例如，以下程序仅保留了姓名和总分在Class数据集中。

```
data class ;
input name$ chinese maths english @@;
keep name total;
total=chinese+maths+english;
cards;
a 82 78 69 b 90 78 89 c 79 86 98
;
```

以上这段程序可以换为以下程序，其结果是一样的，读者可以提交这段程序，自行检验。

```
data class ( keep= name total) ;
input name$ chinese maths english @@;
total=chinese+maths+english;
cards;
a 82 78 69 b 90 78 89 c 79 86 98
;
```

2. DROP语句

在DATA步中，DROP语句用来规定哪些变量不包含在正被创建的SAS数据集中。该语句适用于正被创建的所有SAS数据集，而且出现在任何地方其作用也相同。当在一个DATA步中同时创建两个或两个以上不同的SAS数据集时，为了区分每个数据集所要包含的特殊变量，需要在该数据集后面使用DROP= variables选项。

例 3.4.7 程序如下：

```
data class (keep=name total) class1 (drop=name total) class2 (drop= sex) ;
input name$ sex$ chinese maths english @@;
drop sex;
total=chinese+maths+english;
cards;
a f 82 78 69 b m 90 78 89 c f 79 86 98
;
```

分析： 该程序提交后，数据集class包含变量name和total，数据集class1包含变量chinese, maths和english，而数据集class2则包含除sex以外的所有变量。

由于DROP语句和KEEP语句的作用恰好相反，所以使用DROP语句时，当然也可以改为使用KEEP语句。实际中为了书写方便，在保留变量较少时，常用KEEP语句；而在删除变量较少时，使用DROP语句。需要说明的是：

注 3.4.3 在KEEP语句或DROP语句中，第一个变量名前不需加“=”，而在数据集后面使用KEEP选项或使用DROP选项时，必须加上“=”；

注 3.4.4 不论是KEEP或DROP语句，还是数据集后面使用KEEP选项或使用DROP选项，变量之间不能使用“，”，而要用空格隔开。

注 3.4.5 KEEP或DROP语句对当前正在创建的所有数据集都起作用，而数据集后面使用KEEP或DROP选项则仅对在它前面的数据集有效，对其他数据集不起作用。

例 3.4.8 试分析下面的SAS代码

```
data d; array x(100); do i=1 to 50;
    do j=1 to 100;
        x(j)=normal(0);
    end;
output;
end;
drop i j; run;

proc print data=d;
run;
```

试问：

- 1). 结合drop,array,output用法，分析其数据集有多少个变量，有多少个记录。
- 2). 若去掉drop语句，则有多少个变量，其中i,j两个变量的取值状况如何？结合output语句以及数据步的运行机制分析其中原因。

§3.4.4 Data步可执行语句（二）——数据集操作(表运算)

一 关系数据库的运算概述

1. 关系 (Relation)

一个关系对应通常说的一张表。元组 (Tuple) 表中的一行即为一个元组。属性 (Attribute) 表中的一列即为一个属性，给每一个属性起一个名称即属性名。关系必须是规范化的，满足一定的规范条件最基本的规范条件：关系的每一个分量必须是一个不可分的数据项。

2. 关系运算

查询、插入、删除、更新

数据操作是集合操作，操作对象和操作结果都是关系，即若干元组的集合存取路径对用户隐蔽，用户只要指出“干什么”，不必详细说明“怎么干”

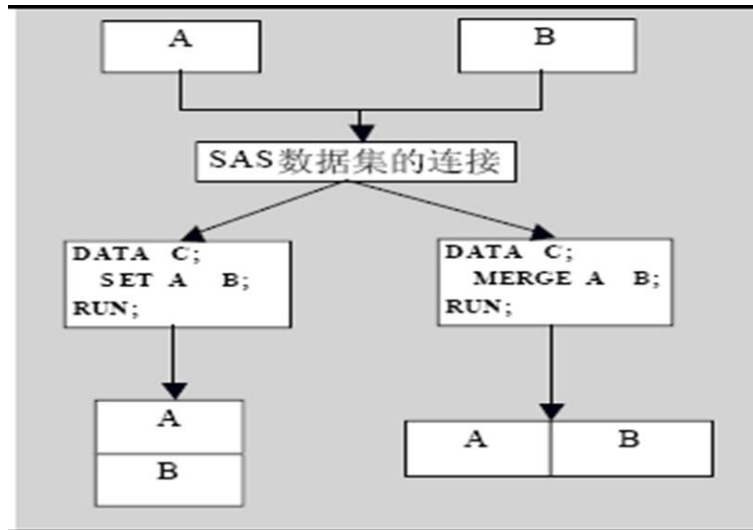


图 3.4: 连接与合并: Set与Merge语句示意图

二 WHERE语句

WHERE语句是一个条件语句，它使得用户从已存在的SAS数据集中把满足条件的观测输出到新的SAS数据集中，它不能用在由CARDS语句构成的DATA步中，也不能用WHERE语句从包含原始数据的外部文件中选择记录。由于WHERE语句执行时，SAS系统不要求从输入数据集中读取所有的观测，所以使用WHERE语句SAS程序将会更有效。WHERE语句在执行SET（连接）、MERGE（合并）、UPDATE（更新）或MODIFY（修改）作业时首先进行选择，然后执行连接、合并、更新或修改。WHERE语句不是可执行语句，因而不能作为IF-THEN语句的一部分。

1. 通常运算和格式

其一般格式是：**WHERE where-expression;**

在DATA步中，WHERE语句必须跟在一个SET，MERGE，UPDATE或MODIFY语句后面，并对SET，MERGE，UPDATE或MODIFY语句中的所有数据集都适用，WHERE表达式中的变量也必须是这些数据集中所具有的。如果DATA步包含WHERE和BY两个语句，那么WHERE语句在BY组被创建之前执行。当创建BY组时，SAS系统对由WHERE语句选择第一个或最后一个观测指定FIRST. Variable或LAST. Variable值为1，而不管那个观测在输入数据集对应的BY组中是第一个还是最后一个观测。

WHERE表达式（where-expression）是由一系列算符和操作数组成的一个算术或逻辑表达式。下列WHERE语句中的表达式都是有效的。

```
where total>240; where score>=60 and score<70; where x; where
x/y; where sex='m'; where sum(chinese, maths, english) >=240;
where sum(chinese, maths, english) < 263; where total in (257 263);
```

与其他SAS表达式一样，在执行WHERE表达式时SAS系统首先对第一个观测计算WHERE后的SAS表达式，之后再决定第一个观测是否执行WHERE语句后的其他SAS语句。如果WHERE表达式运算的结果为真，则SAS系统紧接着执行此后的其他SAS语句；

如果WHERE表达式运算的结果为假，则SAS系统将不执行此后的其他SAS语句，而是返回到DATA步的开始继续处理第二个观测，直到所有观测都被处理为止。

注意，数值变量的名字可以单独出现在WHERE表达式中，如where x语句中的变量x。当运行WHERE语句时，SAS系统判断该变量值是否为0或缺省。如果该变量值是0或缺省，则该表达式运算结果为假，否则为真。

例 3.4.9 给定下面程序：

```
data class ;
    input sex$ chinese maths english @@;
    cards;
    m   82   .   69   f   90   0   89   m   79   86   98
    ;
data class1;
    set class;where maths;
run;proc print;run;
```

程序说明：上述这段程序SAS系统首先创建了一个名为class的数据集，其中，第一个观测变量maths值为缺省,第二个观测其值为0，第三个观测其值为86。接下来，SAS系统利用set语句又创建了一个名为class1的SAS数据集。该数据集利用where maths条件语句选取数据集class中的maths值为非0或缺省的所有其他观测,由此不难理解如下输出结果。

注意，由于WHERE条件语句不能直接用在由原始数据创建的SAS数据集中,所以必须使用SET语句重新创建一个SAS数据集。下面将会看到，这是与IF条件语句重要的区别之一。

2. 特有的运算符

- Between-And算符

该算符选择变量值落在某个范围内的观测，范围的界限可以是常数或表达式。

其一般格式为：**WHERE Variable Between value and value;** par 如：WHERE score Between 60 and 100; 即选择score（分数）在60和100之间的观测；WHERE profit Between 1000000 and 2000000; 即选择profit（利润）在1000000和2000000之间的观测。

- Same-And算符

该算符的意义是除满足已选的条件外，还满足And后的条件的观测。

例 3.4.10 给定下面程序：

```
data class ;
    input sex$ chinese maths english @@;
    total=chinese+maths+english;
    cards;
    m   82   78   69   f   90   78   89   m   79   86   98
    ;
data class1;
```



```

set class;
where total> 180;
where same and total<240;
run;
proc print data=class1;
run;

```

三 连接操作——set语句

SET语句从一个或几个已存在的SAS数据集中读取观测形成一个新的SAS数据集。由于数据的读取是一个观测接着一个观测读取的，因此当读取第一个观测后，如没有其他语句则继续读入第二个观测，直至读完数据集中的所有变量和观测；当有其他语句时，SAS系统根据语句的特点决定是先执行语句后读取观测，还是先读取数据后执行语句。例如，对WHERE条件语句是先执行WHERE条件，然后把满足条件的观测或变量读入到新的数据集中；而IF条件语句则先读取所有的数据，然后再执行IF条件语句。参见子集IF语句和WHERE语句的比较。

1. SET语句的一般格式

SET语句的一般格式为：

```
SET <data-set-name-1< (option-1) >> ...<data-set-name-n< (option-n) >>;
```

由SET语句的一般格式可以看出，SET语句后面可以跟多个数据集，而且数据集后面还可以有选项。注意，如果数据集后面跟选择项，则必须用圆括号（“（）”）括起来。例如下面写法都是正确的。

SET CLASS; 读取数据集CLASS中的数据。

SET CLASS (DROP=SEX FIRSTOBS=5); 删去数据集CLASS中的变量SEX，并从第5个观测开始读取。

SET CLASS1 CLASS2 (KEEP=NAME SEX TOTAL AVE); 将数据集CLASS1与数据集CLASS2中保留的变量NAME SEX TOTAL AVE进行连接。

2. SET语句的用法

SET语句一般用在以下几种场合：

(1). 调用一个已存在的SAS数据文件。即把已存在的SAS数据文件调到当前状态，以便对数据进行其他处理。

(2). 对已存在的SAS数据集进行修改，补充。如原数据集CLASS中含有三科成绩变量chinese maths english的10个观测，现在要统计出每个同学的总成绩和平均成绩，并仍作为CLASS，SAS程序为：

```

data class;
set class;
total=chinese+maths+english;
ave=total/3;
run;

```

(3). 生成已存在的SAS数据集的子集。这有两种情况：一是利用KEEP、DROP选项或语句生成部分变量子集；二是利用WHERE或IF条件语句生成部分观测子集。例如：

```
DATA SEX;  
SET CLASS (KEEP=SEX) ;  
RUN;
```

上面这段程序生成仅含有CLASS数据集中的性别子集。这段程序也可以写为：

```
DATA SEX;  
SET CLASS;  
KEEP SEX;  
RUN;
```

下面这段程序将生成总分在270以上的观测子集。

```
data class1;  
set class;  
total=chinese+maths+english;  
if total>=270 then output;  
run;
```

(4). 把几个已存在的SAS数据集连接成一个新的数据集。

数据集的连接是把两个或两个以上的数据集的所有观测连接在一起，从而形成一个新的数据集的过程。一般来说，如果没有其他特殊规定，SAS数据集的连接过程，可以用以下的图形直观地表现出来，见图

但实际连接过程中，由于对各SAS数据集的观测的排列要求不同，加上各SAS数据集可能存在不同的变量，从而形成以下三种连接方式：

a. 相同变量顺序连接。

这是最简单、应用最多的一种连接方式。在这种方式下，各SAS数据集含有相同的变量。连接的方式是：按各数据集的先后顺序把所有观测连接在一起，即使两个观测完全相同也应按顺序连接起来，因此新数据集的观测数目是所有这些数据集观测数目之和。例如，某系一年级有三个班，期末考试成绩报告单分别以班级编制成三个数据集class1、class2、class3。现要把三个数据集合并成一个数据集grade，以形成年级期末考试结果予以存档。显然，这三个数据集都具有相同的变量（实际中，通常含有学号、姓名、性别、各科成绩、总成绩、平均成绩）。利用set语句，该合并可用以下SAS语句形成：

```
data grade;  
set class1 class2 class3;  
run;
```

b. 不同变量顺序连接。

该种连接方式与相同变量顺序连接方式基本相同，所不同的是由于连接的几个数据集含有不同的变量，所以连接后的变量个数为所有不同变量之和。连接的过程为：从一个数据集中读的观测对其他数据集中也具有相同变量的观测直接连接起来，而没有定义的变量，其值为缺失值。

四 合并操作——merge语句

MERGE语句把两个或两个以上的SAS数据集合并为一个新的SAS数据集，其观测是由合并的数据集中的观测合而为一形成的。其一般格式为：

```
MERGE data-set-name-1<(option-1)> data-set-name-2<(option-2)>  
....<data-set-name-n<(option-n)>>;
```

由MERGE语句的一般格式可以看出，MERGE语句后面至少应跟2个或2个以上数据集，这一点与SET语句有点不同。关于数据集后面的选项，与SET语句基本相同。

MERGE语句的用法，通常有以下两种：

1. 一对一合并

当数据集合并时，若MERGE语句后面没有跟BY语句，此时的合并称为一对一合并，其合并过程为：按MERGE语句后面数据集的排列顺序，把第一个数据集中的第一个观测与第二个数据集中的第一个观测,....,第n个数据集中的第一个观测合并成新数据集中的第一个观测；接下来，把第一个数据集中的第二个观测与第二个数据集中的第二个观测,,第n个数据集中的第二个观测合并成新数据集中的第二个观测；以此类推，直到所有观测被合并为止。合并时，需要注意以下几点：

(1) 合并后新数据集的观测总数为参加合并的数据集中观测的最大值。

(2) 在合并时，如果某数据集已没有观测，则其对应的变量值以缺失值代替。

(3) 在合并时，如果某几个数据集有共同变量，则合并后的新数据集仅含有一个该变量，其对应的值为列在MERGE语句最后一个含有该变量的数据集中的该变量观测值。

2. 匹配合并

合并时，若MERGE语句后跟BY语句，此时的合并称为匹配合并。为了进行匹配合并，每一个数据集至少有一个共同变量，而且必须按共同变量事先排序。用于排序的变量称为匹配变量，通过BY语句排序后形成的每一个组称为BY组。匹配合并实际上就是BY组之间的一对一合并。在具体合并过程中应注意以下几点：

(1) 对每个BY组，新数据集中对应的BY组的观测个数是各合并数据集相应BY组观测的最大值。

(2) 对不同的BY组，SAS系统首先处理那些BY值较小的观测，然后再处理BY值较大的观测。

(3) 在同一BY组中，按MERGE语句后面数据集的排列顺序，把各数据集对应BY组中的第一个观测进行合并，形成新数据集中对应BY组的第一个观测；接下来，把各数据集对应BY组中的第二个观测进行合并，形成新数据集中对应BY组的第二个观测；以此类推，直到所有观测被合并为止。需要注意，在同一BY组的合并过程中，如果某个数据集在某一BY组中没有观测，则按缺省值处理。如果某一数据集含有多个具有相同BY值的观测，则合并时输出所有这些观测，其他数据集对应的BY组，如果没有相应的观测，则取该BY组最后一个观测作为当前观测与之合并。如果某个变量同时出现在几个数据集中，但是这个变量又不是匹配变量，则在新的数据集中，这个变量只出现一次。在这种情况下，如果一个BY组中，各个数据集只有一个观测，则此变量值即为最后一个数据集含有该变量的观测值；如果在一个BY组中，有多个观测，则在新的数据集中此变量值是在MERGE语句中最后一个对BY组提供信息的哪个数据集相应的观测值。

例 3.4.11 给定程序:

```

data A;
input name$ sex$ @@;
cards;
benjim m   rose f david m john m mayers f mayers m
;proc sort;by name;run;
data B;
input name$ computer physical maths english @@;
cards;
rose      89   65   76   87   david   78   86   78   98
david     87   69   77   90   john    90   65   76   87
mayers    89   67   78   80   star     66   81   75   94
; proc sort;by name;run;
data AB;
merge A B;by name; run;
proc print; run;

```

上面这段程序对前面的两个数据集稍微作了点修改。可以看到，数据集A，B都含有变量name，并按该变量对两数据集进行了排序，然后把它们合并成一个新的数据集AB。排序及合并结果如下(略)：

§3.4.5 Data步数组语句

在数据处理过程中，常常会遇到像向量或矩阵这样的组合型数据。这类组合型数据往往具有同一属性，或者是数值型的，或者是字符型的。对于这样一类具有同一属性的数据或变量，SAS系统提供了具有这类数据变量特点的数组语句-ARRAY语句，用以定义一组变量为某个数组的元素。此后，当DATA步中的其他SAS语句需要使用这组变量时，SAS系统可以使用这个数组的元素代替这组变量。

一 显示下标ARRAY语句

显示下标ARRAY语句用以定义数组变量。它规定了数组变量的名字，数组元素的个数、属性、起始值。数组经常出现在DO语句中，用以循环执行这组数据。显示下标ARRAY语句的一般格式为：

```
ARRAY array-name{subscript}<$><length><<array-elements><initial-values>>;
```

ARRAY语句中参数的含义：

(1) array-name (数组名)：规定数组变量的名字，并且不能与同一个DATA步中的其他SAS变量名相同。

(2) Subscript (下标)：它可以是一个*号、或一个数字、或表示该数组元素的个数和范围，而且必须用括号括起来。括号可以是左大括号 ({ })、中括号 ([]) 或小括号 (())。下标具有以下三种格式：

格式1: dimension-size-1<,...dimension-size-n>

指出数组中每一维元素的个数。如数组为多维的，每一维之间必须用逗号 (,) 隔开。例如，一维数组可以如下定义：

```
array economic{4}cost price profit income;
```

该语句定义了一个名为economic一维数组、变量名分别为cost, price, profit, income的四个变量。

二维数组可以如下定义：

```
array score{40,7} x1-x280;
```

该语句定义了一个名为score二维数组、变量名分别为 X_1, X_2, \dots, X_{280} 共280个变量。该二维数组是按顺序从左上角到右下角逐行读入，用矩阵可表示为：

$$\begin{bmatrix} X_{11} & X_{11} & \cdots & X_{11} \\ X_{21} & X_{22} & \cdots & X_{27} \\ \vdots & \vdots & \ddots & \vdots \\ X_{40,1} & X_{40,2} & \cdots & X_{40,7} \end{bmatrix} \quad (3.2)$$

格式2: <lower:>upper<,...<lower:>upper>

此格式规定数组中每一维的上下界。lower表示一维数组的下界，而upper表示一维数组的上界。如果没有规定下界，则表示数组下界为1。如：一维数组语句array economic{4}cost price profit income中的economic{4}也可以写为economic{1: 4}，二维数组语句array score{40,7} X1-X280中的score{40,7}也可以写为：score{1: 40,1: 7}。二维数组语句array test{3: 4, 3: 6} t1-t8；中的变量t1由数组元素t{3, 3}代表，而t8则由t{4, 6}代表。

格式3: {*}

该格式表示SAS系统通过数组中变量的个数来确定下标，当使用星号{*}格式时，语句中必须包含array-elements（数组元素）。当用_TEMPORARY_（临时）数组，或者定义多维数组时，不能使用星号{*}。

显示下标ARRAY语句格式中的\$，表示数组中的元素是字符。如果这些元素以前已经定义为字符，那么这个\$可以省去。Length：规定数组中元素的长度

array-elements：定义变量的名字。如没有该选择项，SAS系统将自动使用数组名和数字1, 2, ..., n来规定变量名。

initial-values：按数组元素的先后顺序给出数组元素的初始值。当数组元素个数恰好与给出的初始值的个数相等时，每一个元素或变量恰好按先后顺序对应一个初始值；当数组元素个数大于给出的初始值的个数时，仍按元素或变量先后顺序把初始值赋给相应的元素，多余的元素或变量以缺省值代替。

例 3.4.12 一维和二维数组可以建立并给定初值：

1). array test{4} t1 t2 t3 t4 (78 90 87 81) ;

2). array test{2,2} t11, t12, t21, t22 (80 79) ;

第一个语句定义了一个数组名为test、变量名分别为t1, t2, t3, t4的一维数组，且四个变量t1, t2, t3, t4的初始值分别为78, 90, 87, 81。第二个语句定义的数组名也是test，但变量名是t11, t12, t21, t22。由于初始值只有80, 79两个，即元素的个数大于初始值的个数，所以SAS系统把80赋给t11, 79赋给t12，而t21, t22以缺省值代替。

二 ARRAY语句中显示下标数组元素的引用

显示下标数组元素的引用比较灵活，它与一般变量一样，凡是可用表达式的地方，都可以使用数组元素。它包括：赋值语句，累加语句，DO语句，IF语句等。需要注意的是：在使用数组元素时，该数组必须已经被定义，亦即定义数组的ARRAY语句必须出现在被引用的数组元素之前。

例 3.4.13 现有某班的2名同学7门课程的期末考试成绩资料，要求用ARRAY语句把2名同学7门课程的成绩及总成绩读入到临时SAS数据集test中。

解： 根据题目要求，程序如下：

```
data test;
input name$ number$ sex$ score1-score7;
array char{3} name number sex;
array score{7};
sum=sum (of score1-score7) ;
cards;
      张三      200111      f      70      98      80      69      90      74      78
      李四      200112      m      76      80      84      82      91      79      83
;

proc print;
run;
```

程序说明：在该段程序中，input语句首先定义变量name, number, sex为字符型变量，而score1, score2, ..., score7为数值型变量，随后定义了两个一维数组，其中第一个数组是字符型的，第二个数组是数值型的。由于第一个数组中的变量名name, number, sex已在input语句中定义为字符型变量，所以在char3后\$可以省略。求总分利用SAS函数sum，程序运行结果如下(略)，该例表明：在一个DATA步中可同时定义多个ARRAY语句。

例 3.4.14 为我国城镇居民2005, 2006, 2007年家庭消费结构资料：利用二维数组按以上数据排列方式把上述资料输出到数据集CONSUMER中，并运用累加语句输出每年的消费总额。

解： 根据题目要求，程序编辑如下：

```
data consumer(drop=i j);
input year1-year3 X01-X08 X11-X18 X21-X28;
array year{3};
array consum{0:2,1:8} X01-X08 X11-X18 X21-X28;
do i=0 to 2;
  do j=1 to 8;
    if consum(i,j)=600.85 then consum(i,j)='.';
    sum+consum(i,j);
  end;
end;
```

```

end;
cards;
2005 2006 2007
2914.39 800.51 446.52 600.85 996.72 1097.46 808.66 277.75
3111.92 901.78 498.48 620.54 1147.12 1203.03 904.19 309.49
3628.03 1042.00 601.80 699.09 1357.41 1329.16 982.28 357.70
;

proc print noobs;
run;

```

我们再编制以下程序进行比较:

```

data consumer(drop=i);
input year X1-X8;
array consum{8} X1-X8;
sum=0;
do i=1 to 8;
    if consum(i)=996.72 then consum(i)='.';
    sum+consum(i);
end;
cards;
2005 2914.39 800.51 446.52 600.85 996.72 1097.46 808.66 277.75
2006 3111.92 901.78 498.48 620.54 1147.12 1203.03 904.19 309.49
2007 3628.03 1042.00 601.80 699.09 1357.41 1329.16 982.28 357.70
;

proc print noobs;
run;

```

由输出结果可以看出第二次编制的程序符合题目要求，而第一次编制的程序则不符合。这是因为第一段程序把每一个数值都赋给了一个变量，从而SAS系统输出结果，数据集consumer只有一个观测，却有28个变量。

三 隐含下标ARRAY语句

隐含下标ARRAY语句的一般格式:

```
ARRAY array-name<{ variable}><length><<array-elements><initial-values>>;
```

可以看出，在隐含下标ARRAY语句中，显示下标ARRAY语句中的数值下标（subscript）被换为下标变量（variable），其他则基本相同。如果用户没有规定下标变量，SAS系统将使用自动变量_I作为下标变量。隐含下标变量的值从1到这个数组元素的个数。

注意：用户规定的下标变量包含在正被创建的SAS数据集中，但是自动变量_I的值却不包含在数据集中。

例 3.4.15 给定如下程序:

```
data test;
input X1-X5 Y;
array t X1-X5;
do _i_=1 to 5 while (t(_i_)<Y);
    output;
    put t{_i_}= Y=;
end;
cards;
1 3 5 7 4 6
0 2 7 6 8 5
;
proc print;
run;
```

解: 程序说明: 上述这段程序利用隐含下标数组语句array和循环DO语句创建了一个名为test的临时SAS数据集, 其中, array语句中的数组名为t, 下标被隐含, SAS系统自动用_i_来代替, 数组t对应的变量X1, ..., X5。循环语句利用while (t(_i_) < Y)来控制循环次数, 由输入的第一个观测值可知, 变量X1, X2, X3的值都小于Y, 而X4的值大于Y, 跳出循环, 所以系统循环三次。Output语句使每次循环都形成数据集中的一个观测, 从而输入时的第一个观测, 输出后成了三个观测。类似可考虑第二个观测。程序运行结果如下。请读者自行考虑当上述程序中的X1, ..., X5的值都小于变量Y时, 其他都不变, 数据集test含有几个观测。

§3.4.6 Data步控制语句

在DATA步中, SAS系统对每个观测是按照每个语句出现的先后顺序依次执行的。有时当你想改变语句的运行顺序或对某些确定的观测跳过一些语句, 就需要使用DATA步中的控制语句。

一 IF语句(条件语句)

在SAS系统中有两种形式的IF语句: 条件IF语句和子集IF语句。子集IF语句不包含THEN子句, 它仅对满足IF条件的那些观测进行处理。

1. 条件IF语句

条件IF语句含有一个THEN子句, 用来执行满足IF条件的那些观测。如果某个观测不满足IF条件, 则THEN子句不被执行, 此时可以使用ELSE语句来执行有关的运算。如果没有规定ELSE语句, 则SAS系统执行紧跟在IF后面的下一个语句。

其一般格式是: **IF condition THEN statement; ;ELSE statement;;**

以上IF语句, SAS系统首先计算IF后面的条件(condition)。如果计算结果为非0, 认为条件成立, 则执行THEN后的语句(statement)。如果计算结果为0或缺省, 认为条件不成立, 则不再执行THEN后的语句(statement), 而是执行ELSE后的语句或紧跟在IF后面的下一个语句。例如:


```
if total>=240 then output class;
if X=3 THEN Y=X;
if status='ok' and type=3 THEN count+1;
if total>=240 then output class;else output class1;
```

注意，（1）IF-THEN-ELSE语句还可以嵌入一个IF-THEN-ELSE语句。例如：

例 3.4.16 程序如下：

```
data class ;
input sex$ chinese maths english @@;
total=chinese+maths+english;
if sex='m' then
    if total>=240 then output;
    else put 'total<240';
else output;
put 'sex=f';
cards;
m 82 78 69 f 90 78 89 m 79 86 98 m 76 56 80
f 72 76 81 f 69 78 91 m 92 71 85
;
proc print;run;
```

分析： 程序说明：由data步创建的SAS数据集class中有一个嵌套if语句。第一个if语句首先判断性别是否为男性，若是男性则进行第二个条件，判断该同学总分是否在240分以上。如果在240分以上，则该观测读入到正创建的SAS数据集中；如果总分在240以下，则该观测不被读入到正创建的SAS数据集中，只是在LOG窗口中记上该男生总分在240分以下。当第一个观测为女性，即第一个条件不成立时，直接进入else语句，输出该观测，并在LOG窗口记上该生为女性。

表 3.10: 由IF嵌套语句所创建的数据集class的数据结构

OBS	SEX	CHINESE	MATHS	ENGLISH	TOTAL
1	f	90	78	89	257
2	m	79	86	98	263
3	f	72	76	81	229
4	f	69	78	91	238
5	m	92	71	85	248

这里put语句输出的结果都在LOG窗口中出现，而在OUTPUT窗口则不出现。

（2）在IF-THEN-ELSE语句中，还可以使用循环DO语句，这将在循环语句中介绍。

2. 子集IF语句

子集IF语句不包含THEN子句，它仅对满足IF条件的那些观测进行处理。

其一般格式是：**IF expression;**

子集IF语句表明：若某个观测使表达式（expression）的计算结果为非0或缺省，则认为条件成立，SAS系统把当前的（statement）观测读入到正被创建的SAS数据集中，并继续执行此后的语句。若某个观测使表达式（expression）的计算结果为0或缺省，则认为条件不成立，SAS系统立即返回到DATA步的开始，对其他观测执行DATA步，该观测也不被读入到正被创建的SAS数据集中，子集IF语句后的语句也将不再执行。例如：

```
if place = 'Shanghai';
if sex='f' ;
if status='ok' and type=3 ;
```

3. 子集IF语句和WHERE语句的比较

子集IF语句和WHERE语句两者都是条件语句，都需要根据跟在其后的表达式计算的结果来判断条件成立与否，在很多情况下两者输出的结果也一样，但两者之间仍有较大差别。

（1）两者之间最大的差别是WHERE语句在观测读入到程序数据向量之前起作用，而IF语句对已经在程序向量内的观测起作用。因此，在选择观测时，WHERE语句往往比子集IF语句更有效。

（2）WHERE语句仅能处理已存在的SAS数据集中的观测，而子集IF语句不仅能处理已存在的SAS数据集中的观测，也可以处理用INPUT语句产生的观测。

（3）WHERE语句不是可执行语句，而子集IF语句是可执行。

例如：把上例中的where maths换成if maths,其得到的数据集的结果是一样的。但有时子集IF语句和WHERE语句即使在条件一样的情况下，得到的结果也不一定相同，参见下面例。

例 3.4.17 用WHERE语句和子集IF语句产生不同的SAS数据集。

```
data a ;
input x y @@;
cards;
1 15 3 78 5 90
;
data b;
input x z @@;
cards;
1 69 2 89 4 98
;
```

（使用WHERE语句）

```
data whereab;
merge a b;
where x>2;
run;
proc print data=whereab;run;
```

（使用子集IF语句）

```
data ifab;
merge a b;
if x>2;
run;
proc print data=ifab;run;
```

分析： 使用WHERE语句的运行结果使用子集IF语句的运行结果

OBS	X	Y	Z	OBS	X	Y	Z
1	4	78	98	1	4	90	98
2	5	90	.				

上述两个数据集之所以出现不同的结果，是由于使用WHERE语句时，先对数据集a, b应用WHERE语句，由此可得到数据集a, b的子集分别为：

WHERE语句下数据集a, b的子集

X	Y	X	Z
3	78	4	98
5	90		

然后执行两子集的合并，从而得到以上输出结果。使用子集IF语句时，是先合并数据集a, b，再执行IF语句。 ■

if语句下数据集a, b的子集

X	Y	Z		X	Y	Z
1	15	69				
2	78	89	→	4	90	98
4	90	98				
合并后的数据集				IF语句下得数据集		

二 DO语句

DO语句是一种循环语句，即在DO 后面直到出现END语句之前的这些语句被作为一个单元处理，称为DO组。对于DO语句，根据需要可以嵌套使用。常见DO语句有以下五种形式：简单DO语句、循环DO语句、DO WHILE语句、DO UNTIL语句、DO OVER语句。

1. 简单DO语句

简单DO语句常常用在IF-THEN-ELSE语句里，用来执行当IF条件成立时的一组语句，在IF条件不成立时，跳出这组语句去执行其他SAS语句。

其一般格式是：**DO；其他SAS语句；END；**

例 3.4.18 用给定程序。

```
data class ;
input sex$ chinese maths english @@;
if sex='m' then
do;
```

```

        total=chinese+maths+english;
        n+1;
        end;
ave=sum (chinese, maths, english) /3;
cards;
m      82      78      69      f      90      78
89      m      79      86      98      m      76
56      80      f      72      76      81      f
69      78      91      m      92      71      85
;
proc print;run;

```

程序说明：上述这段程序SAS系统首先创建了一个包含性别在内的名为class的SAS数据集。IF语句首先判断性别是否为男性，若是男性，再执行其中的DO组；若是女性，则直接执行IF后的赋值语句：ave=sum (chinese, maths, english) /3。由于女性同学没有执行IF条件中的DO组，所以女性同学的总分为缺省，同时累加语句也没有执行，从而n值不变。

表 3.11: 由循环DO组所创建的数据集CLASS的数据结构

OBS	SEX	CHINESE	MATHS	ENGLISH	TOTAL	N	AVE
1	m	82	78	69	229	1	76.3333
2	f	90	78	89	.	1	85.6667
3	m	79	86	98	263	2	87.6667
4	m	76	56	80	212	3	70.6667
5	f	72	76	81	.	3	76.3333
6	f	69	78	91	.	3	79.3333
7	m	92	71	85	248	4	82.6667

2. 循环DO语句

循环DO语句是指DO-END之间的语句被重复执行的语句，其一般格式为：

```

DO index-variable =spacification-1<,...spacification-n>;
其他sas语句;
END;

```

关于循环DO语句一般格式的几点说明：

- (1) index-variable: 控制变量，它的值控制着执行情况 & 执行次数。
- (2) spacification: 说明项，说明变量的起止值，规定变量的增加值。其一般格式为：

```

start <to stop><BY increment><WHILE|UNTIL (expression) >

```

这里start是循环控制变量的起始值，当同to stop或BY increment一起使用时，必须是数值或产生数值的表达式。循环DO语句从start开始执行，其值在第一次循环之前被计算。当没有使用to stop或BY increment时，start可以是数值常数或字符常数。例如：

do i=3; 表示DO组循环一次；

do i=1, 3, 5, 7; 表示DO组循环四次；

do month='JAN', 'MAR'; 表示DO组循环二次。

to stop: 循环控制变量的终止值，它可以是数值或产生数值的表达式，它的值在每次循环执行前被计算。当start和stop一起使用时，循环地执行DO组中的语句直到循环控制变量的值大于stop的值为止。

BY increment: 规定循环控制变量每循环一次后的增加量，它一般与start, to stop一起使用。如没有此增量，则控制变量的值每次仅增加1。很显然，如果增量大于0，则start是该循环的下界，而stop则是该循环的上界。如果增量小于0，则start是该循环的上界，而stop则是该循环的下界。

WHILE (expression): 规定expression (表达式) 在每次循环执行前先计算，然后根据表达式的真、假决定是执行还是不执行循环DO组。

UNTIL (expression): 规定expression (表达式) 在每次循环执行后再计算，于是在DO组内的这些语句被重复执行直到表达式是真的为止。

例 3.4.19 利用循环语句和随机正态函数产生参数为5的随机数50个。

解: 根据题目要求，程序如下:

```
data a (drop=i j z) ;
do i=1 to 50 by 1; y=0;
  do j=1 to 5;
    z=normal (0) ;
    y=y+z*z;
  end;
  output;
end;
proc transpose out=transa;run;
proc print noobs;run;
```

程序说明: data语句定义了一个名为a的SAS数据集,由于选择项drop=i j z, 所以该数据集不含有变量i, j, z。循环语句中嵌套了一个循环语句, 其中, 第二个循环语句产生参数为5的分布随机数, 这里利用了是由标准正态分布经平方得到这一性质; 第一个循环语句产生50个随机数。为了使50个随机数按行排列, 这里使用了转置过程, 打印过程使用了选项noobs, 表明不输出观测序号。程序运行结果如下: ■

3. DO WHILE语句

该语句规定当WHILE后的表达式运行的结果为真时, 重复地执行DO组中的语句。

其一般格式为: **DO WHILE (expression) ;**

这里的表达式 (expression) 与前面遇到的表达式是一样的, 值得注意的是: 其值是在每次循环开始前被计算。

表 3.12: 由循环语句和随机正态函数产生的50个分布随机数

6.80274	3.02847	3.40320	2.82531	2.87888	6.67964	8.39475	2.40041	4.27569	2.26806
4.22708	5.98785	3.40582	5.10209	3.69296	2.58844	2.13481	4.82308	0.87321	2.59291
3.57369	12.1011	12.7229	12.6032	11.3558	14.1921	6.93884	8.03782	3.98001	1.73402
3.15601	4.29460	14.9750	4.70506	5.47378	3.41526	6.15517	4.62025	1.92610	2.52117
4.80457	4.91229	4.07016	3.66006	5.06776	6.86402	6.74476	8.27617	4.93147	3.54450

4. DO UNTIL语句

该语句规定直到UNTIL后的表达式运行的结果为真时，循环结束。

其一般格式为：**DO UNTIL (expression) ;**

这里的表达式 (expression) 与DO WHILE语句中的表达式是一样的，值得注意的是：其值是在每次循环结束后被计算，亦即DO组中的语句至少被执行一次。

例 3.4.20 程序如下

```
data class ;
input sex$ chinese maths english @@;
do i=1 to 10 by 2 while (n lt 2) ;
total=chinese+maths+english;
n+1;
end;
ave=sum ( chinese, maths, english ) /3;
cards;
m   82   78   69   f   90   78   89   m   79   86   98   m   76   56   80
f    72   76   81   f   69   78   91   m   92   71   85
;
proc print;run;
```

程序说明：这段程序利用data步创建了一个包括循环语句、赋值语句和累加语句在内的名为class的SAS数据集。其中，循环变量的起始值为1，终止值为10，步长增量为2，并满足累加变量n小于2，从而当程序读入第一个观测执行到该循环语句i=1时，SAS系统首先对n和2进行比较。由于n开始为0（系统默认），小于2，所以系统执行该循环语句。经循环一次后，i增加2为3，累加变量n也加1变为1，此时n与2比较仍小于2，所以继续执行一次循环语句，之后i变为5，而n也增加到2，此时n与2再进行比较条件已不成立，退出循环，转而执行赋值语句：ave=sum (chinese, maths, english) /3。当系统执行完第一个观测重新读入第二个观测时，由于n等于2没变，所以直接跳过循环语句，执行此后的赋值语句，由此得到以下输出结果：

5. DO OVER语句

DO OVER语句是用来处理具有隐含下标数组元素的循环语句。其一般格式为：

DO OVER array-name;

表 3.13: 由循环DO组所创建的数据集CLASS的数据结构

OBS	SEX	CHINESE	MATHS	ENGLISH	I	N	TOTAL	AVE
1	m	82	78	69	5	2	229	76.3333
2	f	90	78	89	1	2	.	85.6667
3	m	79	86	98	1	2	.	87.6667
4	m	76	56	80	1	2	.	70.6667
5	f	72	76	81	1	2	.	76.3333
6	f	69	78	91	1	2	.	79.3333
7	m	92	71	85	1	2	.	82.6667

其他SAS语句;

END;

该语句对数组中的每个元素自动地执行DO组中的每个语句。具体实例见array语句。

§3.5 Proc步机理、功能与常见语句

§3.5.1 Proc步机理与功能

§3.5.2 Proc步通用语句

一 PROC语句

其一般格式为: **PROC proc-name <options>;**

该语句表示PROC步的开始和调用的过程类型。其中, proc-name表示用户想要使用的SAS过程名字;

options (选择项) 规定关于这个过程的一个或几个选项。由于过程性质不同, 其规定的选项也有所不同, 但是下面三类选项却是共同的:

keyword (关键词): 是该过程进一步要求的单个关键词。keyword=

value: 规定关键词和值, 其中, value可以是数值或字符串。

keyword=SAS-data-set: 规定输入或输出的SAS数据集。

例如: Proc means data=class maxdec=3 sum mean range; 表示系统调用均值过程, 并计算数据集class各数值变量的和、均值和极差, 并保留3位有效小数。

Proc plot hpct=50 vpct=33; 表示在高为33%, 宽为50%的纸上绘制正在处理的数据集的曲线图。

由于问题的不同, 分析的方法不同, SAS系统所提供的非常多的过程中的选择项也非常多, 需要针对不同的过程加以区别, 这为学习SAS带来了一定的困难。建议读者在学习有关过程时, 除了要记住有关过程名外, 要特别注意这些选择项, 这是学好SAS系统的关键所在。

二 VAR语句

其一般格式为：VAR variables-name;

该语句表示哪些变量将被分析和计算，其任意有效的表达形式都可以使用。

例如：Var weight height; Var X1-X15 Y;

三 BY语句和CLASS语句

BY语句的一般格式为：BY variables-name;

该语句表示过程按给出的变量进行分组并分析。注意，当要对一个数据集按某一变量进行分组分析时，SAS系统要求首先应按该变量进行排序，然后再按该BY变量进行分析。

例如：假定数据集sasuser.class已按sex排序，则程序Proc print data=sasuser.class; by sex; run; 提交后将产生男、女分开输出的列表。

例 3.5.1 下列数据(见SAS程序数据区)来自某市房地产公司的资料，试按房屋类型和每一房屋类型下卧室个数的多少计算其平均售价。

解： 根据题目要求，程序编辑如下：

```
data house;
input style$ sqfeet bedrooms baths price;
cards;
RANCH      1250      2      1.0      64000
SPLIT      1190      1      1.0      65850
CONDO      1400      2      1.5      80050
TWOESTORY  1810      4      3.0     107250
RANCH      1500      3      3.0      86650
SPLIT      1615      4      3.0      94450
SPLIT      1305      3      1.5      73650
CONDO      1390      3      2.5      79350
TWOESTORY  1040      2      1.0      55850
CONDO      2105      4      2.5     127150
RANCH      1535      3      3.0      89100
TWOESTORY  1240      2      1.0      69250
RANCH      720       1      1.0      34550
TWOESTORY  1745      4      2.5     102950
CONDO      1860      2      2.0     110700
;

proc sort;
by style;
run;

proc means mean;
var price;
```



```
by style;
run;

proc sort data=house;
by style bedrooms;
run;

proc means mean;
var price;
by style bedrooms;
run;
```

程序说明：DATA步创建了一个由房屋类型、建筑面积、卧室个数、浴室个数和房价五个变量组成的数据集house。第一个sort排序过程按房屋类型进行排序，随后的均值（means）过程计算各种房屋类型的平均房价，均值过程中的选择项mean表示仅计算房价的平均值；第二个sort排序过程先按房屋类型进行排序，然后再按卧室个数进行排序，即在每种房屋类型下再按卧室个数进行分组，对应的均值过程也按相应的组计算平均房价。

CLASS语句的一般格式为：CLASS variables-name;

该语句表示对指定的变量进行分组。当过程中使用该语句时，SAS系统按指定的分组变量（variables-name）进行分组分析。值得注意的是，使用CLASS语句进行分组分析，分类变量不需要事先排序，这一点与使用BY语句是不同的。

例 3.5.2 在上例的第一个均值过程，如用CLASS语句进行分组，则程序为：

```
proc means mean;
var price;
class style;
run;
```

从输出结果可以看出，利用BY语句和利用CLASS语句进行分组，其输出结果基本相同

四 ID语句

ID语句的一般格式为：ID variables-name;

该语句表示用给出的变量来识别观测，这里的variables-name为识别变量名。当该语句与PRINT语句一起使用时，输出的观测用ID变量的值来识别，而观测的序号则不会输出。

注意，使用ID语句识别观测，使用的ID变量值要求与观测一一对应。

五 MODEL语句

MODEL语句一般格式为：MODEL dependent-variables-name=independent-variables-name;options;

该语句表示过程中使用的模型结构，等号左边的变量是因变量，等号右边的变量是自变量，根据过程的特点来选。该语句通常用在回归分析和方差分析中。

例如：model Y=X1 X5; model Y1 Y2=A B C;

六 WEIGHT语句

WEIGHT语句一般格式为：WEIGHT variable-name;

该语句表示用给定的变量值对相应的这个观测中其他变量进行加权。该变量值应大于0，若小于0或缺省，取该值为0。WEIGHT语句通常用在每个观测的重要性不同的情形下。

例如，当询问未来市场走势时，很显然，专家对未来的判断要比一般普通大众的判断重要得多，这时就可用WEIGHT语句对专家给出的预测值进行加权。

七 FREQ语句

FREQ语句一般格式为：FREQ variable-name;

该语句中变量（variable）的值是相应的这个观测中其他变量值出现的次数。若变量的值小于1或缺省，相应的观测不参加计算统计量；若这个值不是正整数，取整数部分。

注意：WEIGHT语句与FREQ语句之间的区别。FREQ变量值表示其他变量相应观测值出现的次数，而WEIGHT变量值则表示对其他变量相应值进行加权，关于这一点可从以下例子中看出。

例 3.5.3 给定下面程序：

```
data q;
input a b @@;
cards;
4 4 5 7 3 4
;proc means;
freq b;
run;
proc means;
weight b;
run;
```

由以上输出结果可以看出，对变量a用变量b作为频数和作为权重，两者的结果是不一样的。首先，两者观测次数不同，前者观测次数为变量b的所有值之和，而后者的观测次数仍是变量a的观测次数；其次，两者虽在均值、最大、最小值上是一致的，但其标准差却不一样。

八 OUTPUT语句

OUTPUT语句给出用该过程产生的输出数据集的信息。其一般格式如下：

OUTPUT <OUT=SAS-data-set> <keyword=names>;

SAS-data-set规定用这个过程产生的输出数据集的名字，如果该选项缺省，SAS系统自动按DATAn的名字命名；keyword=names规定在这个新数据集中同关键词联系的输出

变量名字，关键词随着不同的过程而变化，但通常是一些描述统计量或者输出到新数据集的其他值。

例如：Proc means ; var X; output out=outmean mean=meanx; run; 此例用过程means计算变量X的均值。关键词mean=规定变量X的均值用名字meanx作为输出数据集outmean中的变量。

§3.6 常见全程语句


§3.6.1 概述


定义 3.11 全局通用语句是可以用在任何地方的SAS语句。这些语句既可以用在数据步（DATA步），也可以用在过程步(PROC步)，甚至还可以单独使用。

§3.6.2 注释语句

定义 3.12 注释语句可以放在SAS程序的任何地方作为程序的说明，或者介绍整个程序的步骤或算法等。其基本格式有两种：

```
* message;
/*message*/
```

 两种格式意义不同：“单行注释”和“多行注释”。前者是语句的标准形式，一般做单行注释；后者则是多行注释，没有语句的影子。

 应用场合

- 1). 信息的长度可任意，但不能包含分号(;)，最后的分号表示信息结束。
- 2). 中间的信息可以包含分号，但这种形式的注释不能嵌套。

例 3.6.1 注释语句应用。

```
proc print data=ResDat.stk000002 (obs=10) noobs;
/*输出前10个观测，不输出观测序号*/
var oppr hipr lopr clpr;
title"股票行情";
run;

proc means data=ResDat.stk000002 ;
*对数据集ResDat.stk000002使用means过程;
var oppr hipr lopr clpr;      /*输出变量oppr hipr lopr clpr的均值*/
run;
```

例 3.6.2 标准SAS程序开头，记录SAS程序信息的注释形式。


```
/*-----*/
/*---- Begin Estimation for Grunfeld's Investment Models ----*/
/*---- See SAS/ETS User's Guide, Version 5 Edition, -----*/
/*---- pages -----*/
/*-----*/
```


§3.6.3 TITLE语句

定义 3.13 TITLE语句规定SAS输出文件和其它SAS输出标题。每一个TITLE语规定一级标题，最多可规定10级标题。语句格式：

```
TITLE<n><'text'|"text">;
```

其中：command规定主机操作系统的命令。

 n紧跟在词TITLE后面(不能有空格)的数字，用来规定标题的级别；text规定标题的内容。

 规定标题的内容一直有效，但可以重新规定或取消。

例 3.6.3 1). 只规定第1和第5级标题的内容时，中间标题为空白。

```
title 'this is the 1th title line';
title5 ' this is the 5th title line';
```

2). 取消所有标题内容。

```
title;
```

3). 取消第3级及以后的所有标题内容。

```
title3;
```


§3.6.4 OPTIONS语句


§3.6.5 X语句

定义 3.14 运行SAS系统时，发布主机操作系统命令。语句格式：

```
X <'command'>;
```

其中：command规定主机操作系统的命令。

 格式中的单引号 “ ” 必须；

 应用场合：常见的主机操作系统命令均可以实现。

例 3.6.4 应用举例。

```
x 'mkdir d:\ResDat1';
libname ResDat1 'd:\ResDat1';
data ResDat1.class;
set ResDat.class;
run;
```


例中，在SAS会话期间用主机操作系统命令创建一个目录。注意：键入EXIT命令退出操作系统返回到SAS会话。


§3.6.6 FOOTNOTE语句

定义 3.15 FOOTNOTE语句在每一页的底部输出一些脚注行。最多可产生10个脚注行。语句格式:

```
FOOTNOTE<n><'text'|"text">;
```

其中: command规定主机操作系统的命令。

 n紧跟在词FOOTNOTE后面(不能有空格)的数字,用来规定脚注的行号;text规定脚注行的内容。

 规定的脚注行内容将输出在所有过程的输出页上,但可以重新规定或取消。

例 3.6.5 1). 规定脚注。

```
footnote '西北大学数学系' ;
```

2). 取消所有已规定的脚注行。

```
footnote;
```


3). 取消第3个及以后的所有脚注行。

```
footnote3;
```

§3.6.7 RUN语句

定义 3.16 RUN语句使SAS程序被执行。语句格式:

```
RUN <CANCEL>;
```

 CANCEL让SAS系统结束当前步的执行。SAS将输出一个信息说明这一步没有执行。但CANECL选项不能阻止包含CARDS或CARDS4语句的DATA步执行。

例 3.6.6 不能省略RUN语句的情况。

```
title 'using proc means';
proc means data=ResDat.class min max;
var age height weight;
run; /*此RUN语不能省略省略RUN语句 */
title 'using proc plot';
proc plot data=ResDat.class;
plot age*height;
run;
```

例中，第一个RUN语句在读第二个TITLE语之前执行PROC MEANS步。如果省略第一个RUN语，SAS系统在它读PROC PLOT语句之后执行PROC MEANS步。这时第二个TITLE语覆盖第一个TITLE语，也就是两个过程的输出都包含了标题'USING PROC PLOT'。所以，这种情况下，第一个RUN语不能省略。

例 3.6.7 使用选项CANCEL。

```
proc means data=ResDat.idx000001;
var clpr X; /*注意数据集中没有变量X */
run cancel;
```

例中，当发现SAS程序有错误不能运行这一段序时，使用选项CANCEL结束当前步的执行。

§3.6.8 LIBNAME语句

定义 3.17 LIBNAME语句定义SAS逻辑库。语句格式

```
LIBNAME libref <engine> <'SAS-data-library'>
<Access=Readonly|Temp>;
LIBNAME libref Clear;
LIBNAME libref | _ All_ List;
```

 三种格式反映了LIBNAME语句的三种用法。其选项说明：

Libref 规定逻辑库

Engine 规定引擎


Sas-Data-Library 规定主机系统下一个有效的物理地址

Access=Readonly—Temp 规定逻辑库为只读或可修改属性

Clear 清除与库标记的联系

All 列出所有逻辑库的属性

List 在Log窗口列出逻辑库的属性。

 应用场合：LIBNAME语句把一个libref(库标记名)和一个目录名联系起来，使用用户可在SAS语句中使用库标记来指示这个目录。

例 3.6.8 1). LIBNAME规定不同引擎的逻辑库。

```
libname SASDB1 tape ' SAS- data-library'; /*规定一个TAPE引擎*/
libname SASDB2 V6 ' SAS- data-library'; /*规定版本为V6引擎*/
libname SASDB3 ODBC ' SAS- data-library' ; /*规定版本为ODBC引擎*/
```

2). 不同引擎的逻辑库数据集的转换。

```
libname ResDatv6 v6 'D:\ResDat';
data ResDatv6.class;
set ResDat.class;
run;
```

3). 对已经存在的逻辑库使用LIBNAME语句联系一个SAS引擎。

```
libname SASDB3 ODBC;
```

4). 一个物理地址联系两个库标记。

```
libname ResDat1 'D:\ResDat';
libname ResDat2 'D:\ResDat';
run;
```

5). 脱离与库标记的联系。

```
LIBNAME libref CLEAR;
```

6). 列出逻辑库的属性。

```
libname ResDat list; /*列出逻辑库ResDat的属性 */
libname _all_ list; /*列出所有逻辑库的属性 */
run;
```

7). 多个物理地址指定一个逻辑库。

```
libname new ('d:\ resdat' 'd:\resstk');
```

8). 多个不同的逻辑库组成一个逻辑库。


```
libname new (resdat resstk);
```


§3.6.9 FILENAME语句

定义 3.18 FOOTNOTE语句在每一页的底部输出一些脚注行。最多可产生10个脚注行。语句格式：

```
FOOTNOTE<n><'text'|"text">;
```

其中：command规定主机操作系统的命令。

 n紧跟在词FOOTNOTE后面(不能有空格)的数字，用来规定脚注的行号;text规定脚注行的内容。

 规定的脚注行内容将输出在所有过程的输出页上，但可以重新规定或取消。

例 3.6.9 1). 规定脚注。

```
footnote '西北大学数学系' ;
```

2). 取消所有已规定的脚注行。

```
footnote;
```

3). 取消第3个及以后的所有脚注行。


```
footnote3;
```


§3.6.10 MISSING语句

定义 3.19 FOOTNOTE语句在每一页的底部输出一些脚注行。最多可产生10个脚注行。语句格式:

```
FOOTNOTE<n><'text'|"text">;
```

其中: command规定主机操作系统的命令。

 n紧跟在词FOOTNOTE后面(不能有空格)的数字,用来规定脚注的行号;text规定脚注行的内容。

 规定的脚注行内容将输出在所有过程的输出页上,但可以重新规定或取消。

例 3.6.10 1). 规定脚注。

```
footnote '西北大学数学系' ;
```

2). 取消所有已规定的脚注行。

```
footnote;
```

3). 取消第3个及以后的所有脚注行。

```
footnote3;
```

§3.7 文件输出系统—ODS

§3.8 宏语言—宏变量与宏

§3.8.1 宏机制 (Macro Facility)

一 宏机制构成及运行逻辑

定义 3.20 SAS宏机制是用于扩充和制作用户化SAS系统的工具;当用户在某个SAS程序中使用宏功能时,这个

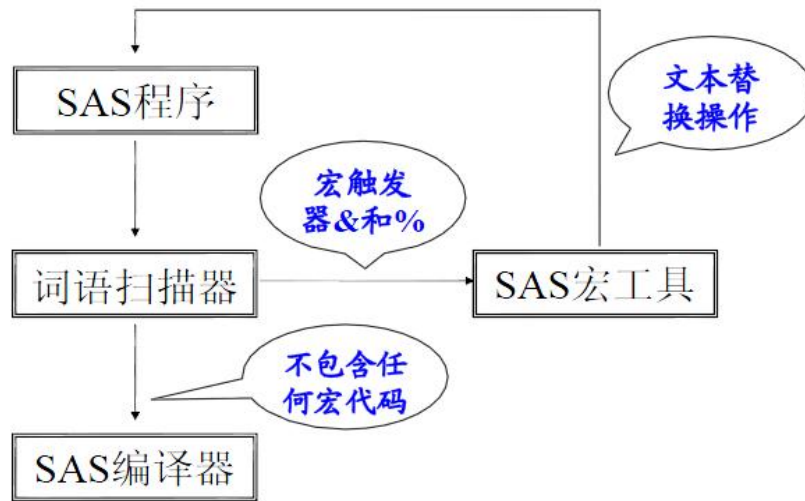


图 3.5: 整个SAS程序的运行逻辑

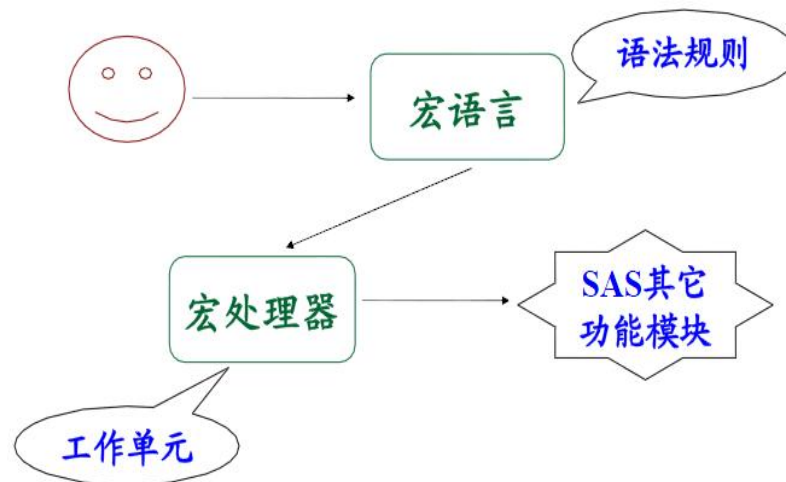


图 3.6: SAS程序中宏的运行逻辑

✎ SAS宏机制的构成：宏处理器（Macro Processor）与宏语言(Macro Language)

(1).宏语言(Macro Language)是用户与宏处理器之间通讯语言。

(2).宏处理器（Macro Processor）是宏程序的翻译器。

✎ SAS宏机制的运行逻辑与整个SAS程序的运行逻辑

✎ 宏程序与基本SAS程序的“碰撞点”——宏变量与宏的单独引用

✎ SAS宏机制的核心思路和引申意义：

核心思路：Macro并不是函数封装的概念，它的核心思路是文本替换

引申意义：在“文本替换”的思路下，产生以下意义

(1). 宏变量的基本功能是替代SAS程序文本，即通过将一段文本赋值给一个宏变量或者作为宏的内容，从而可以通过引用宏和宏变量达到使用这段文本的效果，这种引用特别是在以下两点功能中得到最好的体现；

(2). 利用宏功能用户可以减少在完成一些共同任务时的必须输入的文本量，主要是修改其中部分的宏变量值(包括全局宏变量值或者宏中宏参数值)；

(3). 并能在能进行有选择性和循环性的数据操作（分支结构和循环结构）。

二 宏语言大意

定义 3.21 宏语言是用户与宏处理器之间通讯语言，它既有一般的程序设计语言的基本特点，同时又有其特别之处。

✎ 宏语言的基本语法要素

(1). 宏变量与宏；

(2). 宏变量、常量、宏函数与表达式；

(3). 一般语句(主要是宏内注释语句)；

(4). 控制结构——循环结构与分支结构

✎ 宏程序与基本SAS程序的前世今生

(1). 平行性：两者可以平行运行，可以有互不干扰的逻辑

(2).交叉点：一般来说，宏程序是为基本SAS程序所用，其交叉点在于对于某个宏或者宏变量的单独引用。这就是现实SAS编程时需要筹划之处，即我们到底需要宏完成什么任务，由此基本任务决定关键的宏（宏变量），而其他宏（宏变量）只是为此关键宏（宏变量）而衍生的。

三 宏处理器与宏触发器（宏标志符）

定义 3.22 宏处理器

✎ 宏处理器的触发与宏触发器

✎ 宏处理器的运行机制与宏变量表(占内存空间)

§3.8.2 宏变量

一 宏变量的定义

定义 3.23 宏变量属于SAS宏语言的一个基本要素，其定义和赋值是同时进行的：

`%let 宏变量名=宏变量值;`

✎ 宏变量的定义与赋值总是同时进行，其语句结构遵循SAS语句的基本规则：以关键词 `%let` 开始，以分号 “;” 结束。

✎ 宏变量名的命名规则：与SAS变量名规则相同。

✎ 宏变量的类型与变量长度：宏变量总是取一字符串，可以是键盘上可以找到的任意字符

(1).一般字符序列:

(2).SAS语句段: 运用宏函数 `%str()`

例 3.8.1 定义一个宏变量:

```
%let DSN=ResDat.class;
```

例中，DSN是宏变量名，ResDat.class为宏变量DSN的值。

例 3.8.2 宏变量的值为一段完整的SAS程序段。

```
%let plot=%str(
proc gplot data=a;
plot clpr*date=1;
symbol1 v=star i=join r=1 c=red;
run;
);
```

例中，必须使用`%STR`函数围住宏变量的值，在以后的程序中可以用`&PLOT`来引用这段程序。

```
%let m=2000;
%let n=1;
data a;
set ResDat.stk000002;
where year(date)=&m and month(date)=&n;
&plot;
title2 "&m 年 &n 月份收盘价时序图";
run;
```

例中，如果不执行宏`&PLOT`步的程序时，可将其设定为空值（`%LET PLOT=;`）。

✎ 宏变量的生命期及相应分类：宏变量以使用范围分为全局宏变量和局部宏变量

(1).全局宏变量：从定义开始生存(占有相应的变量值内存)，到SAS程序终止（上述定义局限于全局宏变量定义）。全局变量可以在SAS对话运行期间使用并且可以在程序的任何地方引用

(2).局部宏变量：生存期=所在宏的生存期(运行范围)（其定义参见宏参数）。局部变量则只能在创建该局部变量的宏中使用,在这个宏之外，这个局部变量就没有任何意义。

✎ 自定义宏变量与自动宏变量

(1).自定义宏变量:

上式局限于此。

(2).自动宏变量:

需要记住相应的分类。

✎ (自定义)宏变量与data步变量的区别:

(1).定义位置不同: (自定义)宏变量可以在除数据行之外的任意位置定义; 而data步变量只能在数据步定义

(2).生命期不同: (自定义)宏变量的生命期开始于定义之时, 与SAS程序同时消亡; data步变量与相应的数据集生命期相同

(3).与数据集的关系不同: (自定义)宏变量的取值不依赖于任何数据集(这点几乎决定了宏程序与基本SAS程序的“平行性”或者说“逻辑独立性”); data步变量总是与某一个数据集联系在一起, 且其取值依赖于正被处理的观测(注意data步的“Loop”原理)。

二 宏变量的引用

定义 3.24 宏变量的引用是SAS宏语言的一个基本的语句:

& 宏变量名

✎ 宏变量引用语句不同于其他语句的特别之处就在于: 没有分号。这个特点是由于宏变量引用语句的“二重功能”决定:

(1). 宏程序内部的引用: 主要是宏变量对宏变量的引用, 以及宏对宏变量的引用。

(2). 基本SAS程序中的单独引用: 这点是宏程序与基本SAS程序的“碰撞点”, 火花由此产生, 同时也产生了替换后出现多个分号的诡异局面而违反基本的SAS程序语法, 因而取消该语句的句尾分号, 不能不说是一步关键处的灵活变通。

例 3.8.3 下面引用宏变量:

```
%let a=ResDat.class;
data a;
set &a ;
run;
```

这个带宏变量引用的程序段, 相当于下面的程序:

```
data a;
set ResDat.class;
run;
```

例 3.8.4 改变宏变量的值。

```
%let m=2000;
%let n=1;
data a;
```

```

set ResDat.stk000002;
where year(date)=&m and month(date)=&n;
proc gplot data=a;
title2 "&m 年 &n 月份收盘价时序图";
plot clpr*date=1;
symbol1 v=star i=join r=1 c=red;
run;

```

例中，改变宏变量的值，如：%LET N=2,3,4,5,...,12可以分别得到12个月的时序图。不过，更好的方法是用宏循环来实现这里的要求。

 宏变量引用有一个位置特点：在SAS字符常数类型中出现时，一般要求出现双引号，而非单引号。

例 3.8.5 宏处理器只能在双引号内进行替代。

```

%let a=january;
data;
put "This is the time series plot for &A";
run;

```

例中，在引号内引用宏变量的值时必须用双引号。因为宏处理器只能在双引号内进行替代。这牵涉到一些语法矛盾之间的协调，此处文本字符类型的规则是铁规，需要单引号，宏语言屈服了。


 宏变量的多次引用及其运行原理

例 3.8.6 多次引用宏变量。

```

%let a=ResDat.class;
data male;
set &a ;
if sex='M';
proc print;
title "SUBSET OF &A";
data female;
set &a ;
if sex='F';
proc print;
title "SUBSET OF &A";
run;

```

 嵌套宏变量引用

例 3.8.7 宏变量的嵌套引用。

```

%let m=2000;
%let n=1;
%let xvar=date;
%let yvar=clpr;
%let plot=%str(
proc gplot data=a;
title2 "&m 年 &n 月份收盘价时序图";
plot &yvar*&xvar=1;
symbol1 v=star i=join r=1 c=red;
run;
);
data a;
set ResDat.stk000002;
where year(date)=&m and month(date)=&n;
run;
&plot;
proc print;
title "&m 年 &n 月份收盘价";
run;

```

三 宏变量值的显示

显示宏变量的最简单方法是使用%put语句，它将文本输出到SAS的日志窗口。

语句格式：

```
%PUT <text | _all_ | _automatic_ | _global_ | _local_ | _user_>;
```

例 3.8.8 显示宏变量的值。

```

data _null_;
%let a=first;
%let b=macro variable;
%put &a !!! &b !!!;
run;

```

或者直接写为：

```

%let a=first;
%let b=macro variable;
%put &a !!! &b !!!;

```

LOG窗口显示：

```
first !!! macro variable !!!
```

从上述代码也可以看出SAS程序的两个特点：

- 1). 宏程序与基本SAS程序的独立性，可以独立执行；
- 2). SAS程序中甚至可以没有数据步或者过程步，甚至完全由SAS宏语言出现单独的宏引用或者宏变量引用即可。不过这种做法的实际意义不大。

四 宏变量模拟带参SAS程序段


这个功能可以说是宏变量以下两个基本功能的复合：嵌套宏变量引用与“任意字符值”特性

§3.8.3 宏


一 宏的定义


定义 3.25 宏是被编辑过的可以从SAS程序中调用的程序。同宏变量一样，一般可以使用宏来产生文本。定义宏的格式：


```
%Macro macro_name;  
text paragraph  
%mend macro_name;
```

 宏的定义更多的类似于函数，其结构以语句“%Macro macro-name;”开始，以语句“%mend macro-name”结束，包含三要素：

- 1). 宏名；
- 2). 宏变量参数；
- 3). “宏体” (Macro Body)

 宏名的命名规则：与SAS变量名规则相同。

 宏变量参数，这是可带可不带的；如果带参数，则可以实现某些“数据驱动”(Data-driven)的功能。这是“宏”之所以倍受SAS专家喜爱的原因之一，详见下小节叙述。

 宏体类型：与宏变量的取值类似，可以取一字符串，可以是键盘上可以找到的任意字符，包括某一段程序。这是与一般的所谓“函数”不同之处，一般函数都有一个返回值(与函数体不同)，而宏起文本替换作用，其值可以说就是宏体本身。

例 3.8.9 定义宏plot。

```
%macro plot;  
New dataset  
%mend plot;  
title "Display data set of %plot";
```


例 3.8.10 定义宏plot。

```
%macro plot;  
proc gplot data=ResDat.stk000002;  
title2 "收盘价时序图";  
plot clpr*date=1;  
symbol1 v=star i=join r=1 c=red;  
%mend plot;
```

二 宏的引用


定义 3.26 宏是被编辑过的可以从SAS程序中调用的程序。同宏变量一样，一般可以使用宏来产生文本用一个百分号(%)加宏名字就可以调用该宏。定义宏的格式：


```
%macro_name
```

 宏的引用与宏变量引用类似，语句不同于其他语句的特别之处就在于：没有分号。这个特点是由于宏引用语句的“二重功能”决定：

(1). 宏程序内部的引用：主要是宏对宏的引用。

(2). 基本SAS程序中的单独引用：这点是宏程序与基本SAS程序的“碰撞点”，火花由此产生，同时也产生了替换后出现多个分号的诡异局面而违反基本的SAS程序语法，因而取消该语句的句尾分号，不能不说是一步关键处的灵活变通。

 宏引用有一个位置特点：在SAS字符常数类型中出现时，一般要求出现双引号，而非单引号。

 宏的嵌套引用：

1). 宏对宏的嵌套引用；

2). 宏对宏变量的嵌套引用。

下面举例说明宏对宏的嵌套引用。

例 3.8.11 分以下几步完成：

STEP 1: 产生数据集宏

```
%macro create;
data temp;
set ResDat.&dat;
if year(date)=&year;
%mend create;
```

STEP 2: 画时序图宏

```
%macro plot;
proc gplot data=temp;
title2 "&pr &year1 时序图";
plot &price*date=1;
symbol1 v=star i=join r=1 c=red;
%mend plot;
run;
```

STEP 3: 宏调用宏

```
%macro analyze(dat, year, pr,price,year1);
%create; /*产生数据集TEMP*/
%plot; /*作图*/
%mend analyze;
```


STEP 4: 运行宏

```
%analyze(stk000002, 2000, 收盘价, clpr, 2000);
run;
```

下面举例说明“宏对宏变量的嵌套引用”的应用案例，主要是“模拟带参”功能。

例 3.8.12 改变宏内的值：其中一个办法就是在宏内嵌套引用宏变量；另外一个办法就是“带参宏”。先以前者为例分以下几步完成：

Step 1: 定义宏plot

```
%macro plot;
proc gplot data=ResDat.&dat; title2 "&pr 时序图"; plot
&price*date=1; symbol1 v=star i=join r=1 c=red;
%mend plot;
```

Step 2: 定义宏变量

```
%let dat=stk000002;
%let pr=收盘价;
%let price=clpr;
```

Step 3: 调用宏plot

```
%plot;
```

Step 4: 改变宏变量的值：

```
%let dat=stk000002;
%let pr=最高价;
%let price=hipr;
```

Step 5: 再次调用宏plot

```
%plot;
run;
```

以上各段程序可以连在一起运行。

例 3.8.13 定义并调用宏plot。

```
%macro plot;
proc gplot data=ResDat.stk000002; title2 "收盘价时序图"; plot
clpr*date=1; symbol1 v=star i=join r=1 c=red;
%mend plot; /*以上定义宏plot*/
%plot; /*调用宏*/
run;
```

对一系列的宏变量引用，如data1, data2, data3, 这一系列中部分文本是固定的名称，而另一部分是变化的数字。这时就可以采取间接引用方式。

例 3.8.14 间接引用宏变量。

```
%let data1=x;
%let data2=y;
%let data3=z;
%macro test;
  %do i=1 %to 3 ;
    %put &&data&i;/*&data&i不能用*/
  %end;
%mend test;
%test
```

LOG窗口显示:

```
1      %let lev1=14;
2      %let lev2=15;
3      %let lev3=16;
4      %macro test;
5          %do i=1 %to 3 ;
6          %put &&lev&i;/*&lev&i不能用*/
7          %end;
8      %mend test;
9      %test
14
15
16
```


注意此例中，让我们更清楚的看见几点宏机制的工作原理：

- 1). 理解间接引用宏变量的工作机制：
- 2). 宏语言存在单独的生命力，例如：刚才这段代码，没有一个data步或者proc步，照样也可以运行，并达到预期效果，说明，宏程序有独立生命力。
- 3). 宏处理器被触发必须出现宏引用或者宏变量引用符号，此时才会出现“替换”操作，没有单独的宏或者宏变量调用，不可能出现“替换”操作；

三 带参宏

宏变量和宏结合在一起是一种强有力的编程方法。但应用起来并不十分方便。其作用有点类似于宏变量的模拟带参功能。此两者常被被“带参宏”取代，即：在宏中使用宏参数。

定义 3.27 宏参数是一种特殊的宏变量，是定义在宏%MACRO语句内的宏变量。

 宏参数赋值方式：

- 1). 创建宏参数时直接赋值。
- 2). 运行宏时赋值。

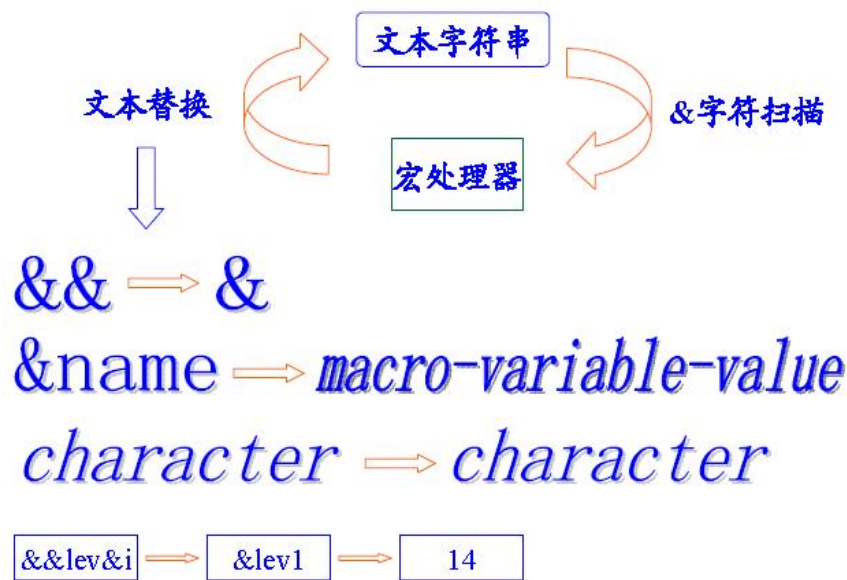


图 3.7: 间接引用宏变量的工作机制

例 3.8.15 带参宏，创建宏参数时直接赋值的使用分为两步：

STEP 1: 创建宏参数。

```
%macro plot(dat,pr,price);
proc gplot data=ResDat.&dat;
title2 "&pr 时序图";
plot &price*date=1;
symbol1 v=star i=join r=1 c=red;
%mend plot;
```

例中，在%MACRO语句括号内定义的宏变量DAT, PR和PRICE称为宏参数。

STEP 2: 通过给宏参数赋值来调用宏。

```
%plot(stk000002, 收盘价, clpr);
```


例中，运行时，宏处理器把第一个值（stk000002）赋给第一个宏参数DAT，第二个值（收盘价）赋给第二个宏变量PR，以此类推

例 3.8.16 带参宏，运行宏时赋值的使用案例：

STEP 1: 创建宏参数。


```
%macro plot(dat=stk000002, pr=收盘价, price=clpr);
proc gplot data=ResDat.&dat;
title2 "&pr 时序图";
plot &price*date=1;
symbol1 v=star i=join r=1 c=red;
%mend plot;
```

```
%plot;
run;
```


 使用宏参数的优点:

- 1). 可以少写几个%let语句;
- 2). 保证该宏参数变量在宏之外的程序部分不被引用;
- 3). 调用宏时并不需要知道这些宏参数的名字, 只要知道相应的取值。

四 宏与宏变量的关系

 区别处: 比较本质的区别有三点

- (1). 一切不同来源于定义形式的不同, 以及由此产生的内在属性不同: 前者是变量, 可以有变量的活动规律(如: 参与运算); 而后者是程序段, 不具有变量性质。
- (2). 封装格局不同: 宏本身可以带参数(局部宏变量); 而宏变量是没有参数的, 虽然这点可以模拟实现。
- (3). 封装物内部的宏语言不同: 这是两者最大的不同。宏变量内部几乎不再能有宏语言, 除了“嵌套引用”下可以在宏变量内部出现宏变量引用(一般不出现宏变量的定义, 这是非常麻烦的, 不妨放在外面); 宏内部则是别有洞天, 下面将要阐述的宏表达式、控制结构等都可以用。

 相似点: 相似处有两点

- 1). 封装物相同: 其均是“机械替代”, “宏变量”与“宏”均可以封装“一切字符”, 包括一般字符和SAS代码段
- 2). 模拟带参程序段这一性质上相似。

§3.8.4 宏变量、常量、宏函数与宏表达式

一 宏函数

二 宏变量与文本表达式

三 比较运算符与逻辑表达式

四 算术运算符与算术表达式

§3.8.5 宏语句与宏控制结构

一 宏语句概述

二 宏控制结构——循环结构

例 3.8.17 定义生成重复文本宏。

```
%macro names(name, number);
%do n=1 %to &number;
&name&n
%end;
%mend names;
```

例中，宏names通过宏参数name，number和宏变量n可以生成一系列的名字。在DATA语句中调用宏NAMES:

```
data %names(dsn, 5);
run;
```

产生下列data步语句:

```
data DSN1 DSN2 DSN3 DSN4 DSN5;
run;
```

三 宏控制结构——分支结构

例 3.8.18 用条件表达式%if-%then定义宏。

```
%macro analyze(getdata, dat, year, pr,price,year1);
%if &getdata=yes %then %create;
%plot;
%mend analyze;
%analyze(yes, stk000002, 2000, 收盘价,clpr,2000);
%analyze(no, stk000002, 2000, 最高价,hpr,2000);
run;
```

四 宏控制结构——其他结构

§3.8.6 Data步接口程序

宏处理器只是在DATA步或者PROC步的编译期间起作用，而不是在执行期间。然而，你也可以编程DATA步，并根据DATA步内的值来创建宏变量，也就是在data步执行期间创建宏变量。

一 data步的接口程序

有两个data步接口程序可在data步执行期间而不是在编译期间用于创建宏变量，指定它们的值，及重新得到它们的值。

1. SYMPUT子程序

它或者创建一个宏变量，其值是来自于DATA步信息；或者指定一个data步值来执行宏变量。

2. SYMGET子程序

它在data步执行期间返回宏变量的值

二 由DATA步的值创建宏变量

假定你想把数据集中的观测个数送到一个Footnote语句中。首先定义宏CREATE如下:

例 3.8.19 基本程序:

```

%macro create;
  data temp;
    set in.permdata end=final;
    if age>=20 then
      do;
        n+1;
        output;
      end;
    if final then call symput('number',n);
  run;
%mend create;

```

在求和(sum)语句中的N用累加计算得出新数据集中的观测个数。在data步执行的最后，symput子程序创建一个名为number的宏变量，其值为N的值。然后用以下语句来定义使用CREATE的宏ANALYZE:

```

%macro analyze(getdata, yvar,xvar);
  %if %upcase(&getdata)=YES %then %create;
  footnote "Plot of &number Observations";
  %plot;
%mend analyze;

```

调用宏analyze:

```
%analyze(yes, income, age);
```

经过“词语扫描”和“替换”之后得到以下SAS程序:

```

data temp;
  set in.permdata end=final;
  if age>=20 then
    do;
      n+1;
      output;
    end;
  if final then call symput('number',n);
run;
footnote "Plot of      26 Observations";
proc plot;
  plot income*age;
run;

```

从以上出现在footnote语句中的观测个数26的位置，发现有一个问题：因为symput子程序用BEST12.的格式输出n的值，故NUMBER的值居右。为了使得改值居左，在SYMPUT子程序中使用left函数:

```
%macro create;
  data temp;
    set in.permdata end=final;
    if age>=20 then
      do;
        n+1;
        output;
      end;
    if final then call symput('number',left(n));
  run;
%mend create;
```

该程序产生的footnote语句为:

```
footnote "Plot of 26 Observations";
```

注意：此例假定第一次调用宏analyze的getdata的值为YES；于是宏create产生数据集TEMP并确定其中的观测个数。

§3.8.7 宏语言的应用实例

- 一 药物代谢动力学处理方法——DSCC的SAS宏实现
- 二 简单商业数据挖掘的案例编程

第四章

数据集的建立

§4.1 方法概述与实例分析

§4.1.1 方法概述

在SAS系统中，数据集是分析的基本对象，任何数据必须以数据集的形式存在才能进行后续的数据处理。Data come in many different forms. Your data may be handwritten on a piece of paper, or typed into a raw data file on your computer. Perhaps your data are in a database file on your personal computer, or in a database management system (DBMS) on the mainframe computer at your office. Wherever your data reside, there is a way for SAS to use them. You may need to convert your data from one form to another, or SAS may be able to use your data in their current form. This section outlines several methods for getting your data into SAS. Most of these methods are covered in this book, but a few of the more advanced methods are merely mentioned so that you know they exist. We do not attempt to cover all methods available for getting your data into SAS, as new methods are continually being developed, and creative SAS users can always come up with clever methods that work for their own situations.

数据的建立，一般分为三类：

(1). 第一类也是最常见的一种途径就是通过SAS程序的Data步实现，属于编程实现的方法；

(2). 第二类就是通过SAS系统的菜单实现，主要包括利用VIEWTABLE新建数据集以及利用SAS ASSIST创建数据集；另外也可以通过子分析系统手工输入实现，常见的是：SAS/Insight系统或SAS/Analysis系统输入。

(3). 最后，还有一类是通过将其他类型或者其他格式的数据文件导入数据集。该类类型转换的方法，也可以通过一些过程步从外部文件读取转化的方法实现。

需要注意的是后两类都是通过菜单实现的，不同的是第二类通过手工输入记录的方式，而第三类则是通过File菜单中的IMPORT数据转换系统支持下，将数据进行转化的结果，是非手工输入的。

§4.1.2 实例分析

下面给定以下问题。下面举例说明之。

例 4.1.1 举两个简单例子：

```
DATA D1;
  INPUT A B;
  CARDS;
  3 5
```



```

7 9
;
RUN;

DATA D2;
  INPUT C D;
  CARDS;
  13 15
  17 19
  21 23
  ;
RUN;

```

§4.2 程序实现方法

§4.2.1 Data步数据集的建立机制

§4.2.2 数据集的建立

一 通过Input语句读取原始数据

通过Input语句在Data步建立一个数据集的方法，包括两种：

(1). 读入CARDS语句后面的数据。

SAS源码 4.1: 数据在作业流

```

1 Data ...;
2 Input ...;
3 (用于DATA步的其它SAS语句)
4 Cards;
5 数据行
6 ;
7 Run;

```

例 4.2.1 建立班级数据集，包含“性别”、“语文成绩”、“数学成绩”、“英语成绩”，如下：

```

data class ;
input sex$ chinese maths english @@;
total=chinese+maths+english;
cards;
m 82 78 69 f 90 78 89 m 79 86 98
;
run;

```

(2). 读入外部数据文件中的数据；

要求外部的数据文件必须是可以操作系统下用TYPE命令打印出其全部内容的ASCII码文本文件，并使用INFILE语句指出从哪一文件中读入数据。

SAS源码 4.2: 数据在SAS数据集

```

1 Data ...;
2 Infile ...;
3 Input ...;
4 (用于DATA步的其它SAS语句)
5 Run;

```

例 4.2.2 如果班级数据是存在于外部文件中class.txt中，则可以如下读入：

```

data class ;
infile "F:\SAS\class.txt";
input sex$ chinese maths english @@;
total=chinese+maths+english;
run;

```

二 通过已有数据集进行表操作

这种数据集建立方式就是利用已经存在的数据集，进行合适的“表操作”。

SAS源码 4.3: 数据在磁盘上

```

1 Data ...;
2 Set | MERGE | UPDATE | MODIFY ...;
3 (用于DATA步的其它SAS语句)
4 Run;

```

(1). set打开一个新数据集

可以用SET打开一个新数据集，并进行变量更新和观测筛选，生成新的数据集。

例 4.2.3

```

data class1;
set class;
where total> 180;
where same and total<240;
run;
proc print data=class1;
run;

```

选出性别变量，去掉其他变量：

```

DATA SEX;
SET CLASS ( KEEP=SEX );
RUN;

```

等价于：

```

DATA SEX;
SET CLASS;
KEEP SEX;
RUN;

```

下面这段程序将生成总分在270以上的观测子集。

```
data class1;
set class;
total=chinese+maths+english;
if total>=270 then output;
run;
```

(2). set打开多个数据集进行连接操作

例 4.2.4 某系一年级有三个班，期末考试成绩报告单分别以班级编制成三个数据集class1、class2、class3。现要把三个数据集合并成一个数据集grade，以形成年级期末考试结果予以存档。显然，这三个数据集都具有相同的变量（实际中，通常含有学号、姓名、性别、各科成绩、总成绩、平均成绩）。利用set语句，该合并可用以下SAS语句形成：

```
data grade;
set class1 class2 class3;
run;
```

§4.3 菜单实现方法

菜单输入数据分为两类，一类是通过子系统输入；另外就是通过外部格式文件的转化。

§4.3.1 子系统输入

— VIEWTABLE

1. 打开 VIEWTABLE

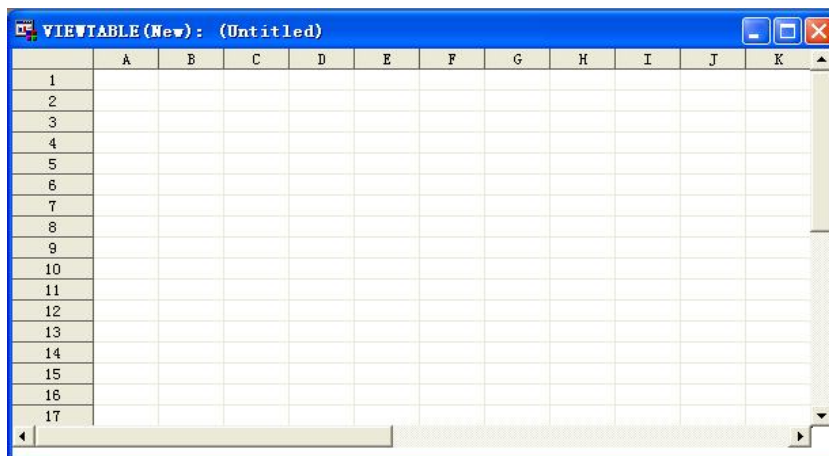


图 4.1: “viewtable” 对话框

The Viewtable window which is part of Base SAS software¹ is an easy way to create new data sets, or browse and edit existing data sets. True to its name, the Viewtable

window displays tables (another name for data sets) in a tabular format. To open the Viewtable window, select Table Editor from the Tools menu. An empty Viewtable window will appear.

Tools \Rightarrow Table Editor, 打开如图 (4.1) 所示的对话框:

This table contains no data. Instead you see rows (or observations) labeled with numbers and columns (or variables) labeled with letters. You can start typing data into this default table, and SAS will automatically figure out if your columns are numeric or character. However, it's a good idea to tell SAS about your data so each column is set up the way you want. You do this with the Column Attributes window.

2. Column Attributes window

The letters at the tops of columns are default variable names. By right-clicking a letter, you can choose to open a Column Attributes window for that column. This window contains default values which you can replace with the values you desire. If you plan to enter date values, then you should choose a date informat so that dates entered will be automatically converted to SAS date values. When you are finished changing column attributes click Close.

3. Entering data

Once you have defined your columns you are ready to type in your data. To move the cursor, click a field, or use tab and arrow keys.

4. Saving your table

To save a table, select Save As ... from the File menu. Select a library, and then specify the member name of your table. The libraries displayed correspond to locations (such as directories) on your computer. If you want to save your table in a different location, you can add another library by clicking the New Library icon. Type in a name for the new library and its path. Then click OK. Specify the member name by typing it in the Member Name field.

5. Opening an existing table

To browse or edit an existing table, first select Table Editor from the Tools menu to open the Viewtable window. Then select Open from the File menu. Click the library you want and then the table name. If the table you want to open is not in any of the existing libraries, click the New Library icon. Type in a name for the new library and its path. Then click OK. To switch from browse mode (the default) to edit mode, select Edit Mode from the Edit menu. You can also open an existing table by navigating to it in the SAS Explorer window, and double-clicking it.

6. Other features

The Viewtable window has many other features including sorting, printing, adding and deleting rows, and viewing multiple rows (the default, called Table View) or viewing one row at a time (called Form View). You can control these features using either menus or icons.

二 SAS/Assist

三 SAS/Analysis或者SAS/Insight

§4.3.2 外部文件转换

Using the Import Wizard, you can convert a variety of data file types into SAS data sets by simply answering a few questions. The Import Wizard will scan your file to determine variable types and will, by default, use the first row of data for the variable names. The Import Wizard can read all types of delimited files including comma-separated values (CSV) files which are a common file type for moving data between applications. And, if you have SAS/ACCESS Interface to PC Files, then you can also read a number of popular PC file types

STEP 1: Start the Import Wizard by choosing Import Data ... from the File menu.

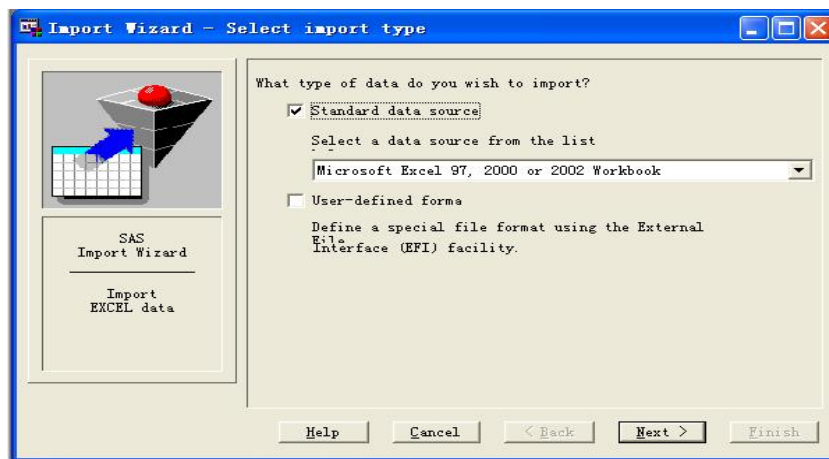


图 4.2: “Import Wizard” 对话框

STEP 2: Select the type of file you are importing by choosing from the list of standard data sources such as comma-separated values (*.csv) files.

STEP 3: Now, specify the location of the file that you want to import.

By default, SAS uses the first row in the file as the variable names for the SAS data set, and starts reading data in the second row. The Options ... button takes you to another screen where you can change this default action.

STEP 4: The next screen asks you to choose the SAS library and member name for the SAS data set that will be created.

If you choose the WORK library, then the SAS data set will be deleted when you exit SAS. If you choose a different library, then the SAS data set will remain even after you exit SAS. There is no way to define a library from within the Import Wizard, so make sure your library is defined before entering the Import Wizard.

STEP 5: In the last window, the Import Wizard gives you the option of saving the PROC IMPORT statements used for importing the file.

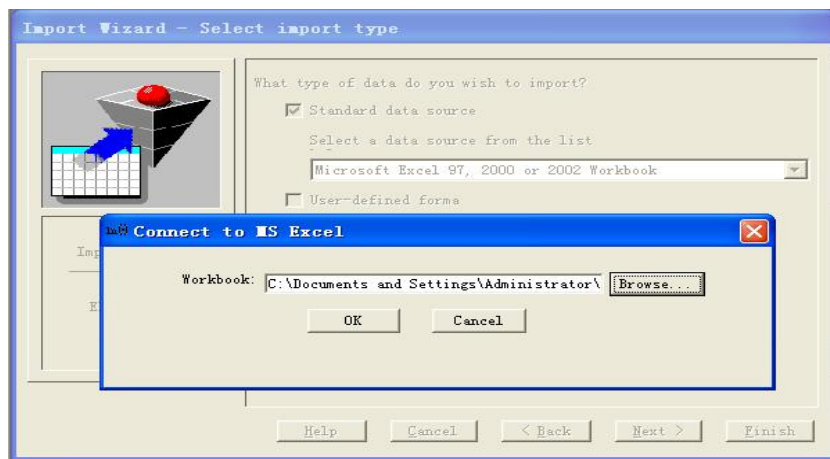


图 4.3: “Import Wizard” 对话框

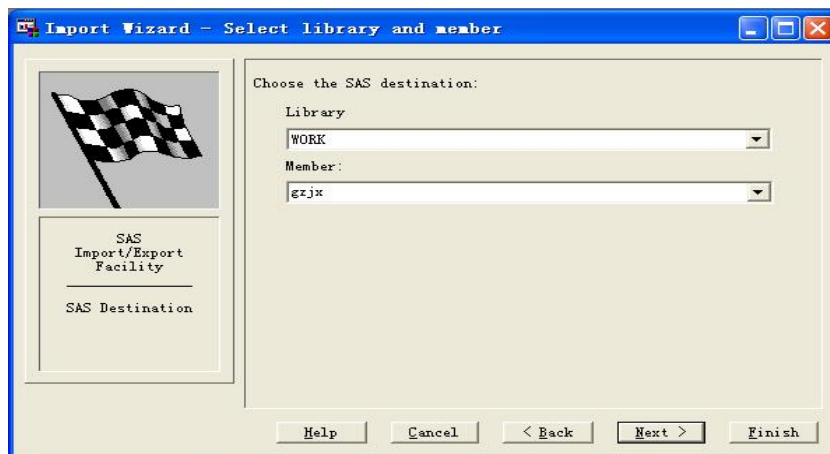


图 4.4: “Import Wizard” 对话框



图 4.5: “Import Wizard” 对话框

STEP 6: For some types of files, the Import Wizard asks additional questions. For example, if you are importing Microsoft Access files, then you will be asked for the database name and the table you want to import. You will also be given an opportunity to enter user ID and password information if applicable.

第五章

抽样调查

§5.1 抽样调查概述

§5.1.1 (直接)数据收集的方法——抽样调查与非抽样调查

数据收集可以直接调查获得，也可以间接获得，前者需要对观测对象进行观察采集，从根源上而言，这是最根本的数据收集方法；后者只是借鉴别人已经获取的数据。

一 数据调查的分类

1. 全面调查与抽样调查

调查是获取数据(资料)的有效手段，它分为全面调查与非全面调查。

定义 5.1 全面调查：是针对总体的每一单元都进行的信息搜集的调查，故也称为普查；**非全面调查：**是那些仅针对总体一部分单元进行的信息搜集的调查，故也称为“抽样调查”(Sampling Survey)。

完整的抽样调查不仅考虑数据收集，也是为特定的统计分析目的相适应的。抽样调查也可以定义为：从构成总体的所有单元中按一定的程序选择一部分单元，并根据这部分单元的特征(调查结果)估计或推断总体特征的调查。

📌 普查的优点与缺点：数据收集和汇总若无任何差错，则普查是可靠的；然而这是很难达到的。

📌 普查优点的前提：

- (1). 总体信息简单，容易调查；
- (2). 总体信息非常重要；
- (3). 具备充足的人力、物力和财力。

📌 抽样的优点：较少的付出，获取较多的信息。

2. 概率抽样与非概率抽样

抽样调查按照是否具有随机性分为：概率抽样(Probability Sampling)与非概率抽样(Non-probability Survey)。

定义 5.2 非概率抽样：是指样本不是按一定的概率抽出，而是抽样者主观抽出或任由受访者自愿进入样本的抽样方法。

📌 常见的非概率抽样方法：

- (1). 判断抽样

也称为立意抽样，指的是根据抽样者主观经验抽取样本。

(2). 典型调查

只对总体中典型(有代表性)的单元进行特征测量的调查

(3). 重点调查:

只对总体中重点的单元进行特征测量的调查

(4). 便利抽样:

根据调查人员的方便, 自行选择入样单元

(5). 自愿抽样:

根据自愿参加的受访者所构成的抽样

✎ 非概率抽样的缺点:

(1). 难以评价样本的代表性

(2). 无法估计抽样误差

(3). 偏倚往往很大(因为往往不具有代表性, 即: 不同分布)

✎ 常见的概率抽样方法:

(1). 简单随机抽样

(2). 分层抽样

(3). 整群抽样

(4). 多阶段抽样

(5). 系统抽样

(6). 不等概率抽样

✎ 概率抽样的优点:

(1). 能够表明一个确定的样本包含哪些单元;

(2). 对每个可能的样本, 都有一个确定的被抽取概率;

(3). 以随机原则抽取样本。这里的需要注意两点: 一是随机原则不等于随便, 指的是不受主观因素的影响, 使每个单元都有一定的概率入选样本, 即要求总体中的每一个基本单元都有一定的(非零)概率被抽中; 另一个就是随机原则不等于等概率原则。

(4). 从样本数据估计总体特征时, 需要考虑该样本被抽中的概率。

二 概率抽样

§5.1.2 总体与样本

一 总体、个体与基本单元

定义 5.3 所研究的对象的全体称为“总体”(Population); 每个研究对象称为“个体”(Individual).

✎ 总体与个体必须都是明确的。

由于实际问题的复杂性, 需要对总体进行必要的修正以便于抽样分析但是尽量保持原样:

(1). 目标总体: 预先设定的理想研究对象的全体;

(2). 抽样总体: 分析的直接对象是抽样总体, 而非目标总体, 但很多情况下两者一致或者近似一致。

☞ 总体的多重理解：注意到冯士雍教授《抽样调查理论与方法》一书以及杜子芳教授《抽样技术及其应用》中刻意强调“抽样中研究的总体与数理统计中总体的概念并不相同”，本人并不认同，而恰恰认为是一体的，无区别的。这本身就是总体概念的三重理解，另外两重理解如下：

- (1). 所研究对象的数量指标的全体
- (2). 随机变量及其概率分布

☞ 个体与基本抽样单位：在抽样调查中，需要有限对象才可进行抽样，这个数量称为“总体规模”(N)；不同于实验观测，只需要进行记录。

- (1). 个体：个体是明确的，但是未必方便抽样；
- (2). 基本单元：实际总体所包含的个体可以是有限的，也可以是无限的。为了抽样方便，总是将总体划分成互不重叠又可穷尽的有限多个部分，每个部分称为抽样单元(Sampling Unit)。事实上，抽样单元可以再加细，因而是可以分级的，最底层的抽样单元称为“基本单元”(Basic Unit)，这是不可再分的、最底层的抽样单位。
- (3). 关系：为了方便抽样而必须产生基本单元；从此，抽样和分析的基本对象是基本抽样单元(unit)，而非个体(individual)，但很多情况下两者一致或近似一致。注意此一区别是非常重要的。

例 5.1.1 举几个总体包含无穷个体，而需要定义不同于个体的基本单元的例子：

- (1). 在粮食农药污染调查中，调查对象(个体)是粮食，这是一种颗粒状的散料，若按颗粒计，它的数量可以看成是无穷的，不好抽样；若是按其存储形式，选取包装(单位：麻袋)或者仓库作为“基本单元”，则将方便抽样。
- (2). 在黄河水水质调查中，调查对象(个体)是水，这是一种溶液分子，要直接调查抽取几乎是不可能的；若是以一瓶(单位：升)为单位，则方便抽取，因而可以作为基本单元。
- (3). 在西安市市民收入调查中，调查对象是每一个市民，方便调查，此时可取基本单元为市民。


二 抽样框与抽样单元

定义 5.4 为了抽样方便，总是将总体划分成互不重叠又可穷尽的有限多个部分，每个部分称为抽样单元(Sampling Unit)。首先，总体先分成若干个规模较大的抽样单元，称为“初级(抽样)单元”或“一级(抽样)单元”(Primary Sampling Unit)；每个初级单元又可包含若干个规模较小的单元，称为“次级(抽样)单元”或“二级(抽样)单元”(Secondary Sampling Unit)；依次可以定义“三级单元”、“四级单元”等，最小一级抽样单元称为“基本抽样单元”。

包含所有抽样单元的分层树状结构或者说“目录性清单”，称为“抽样框”(Sampling Frame)，抽样框中的抽样单元必须是有序的，便于编号抽样。

☞ 抽样单元的基本特性：由基本单元组成，构成总体的一个分割，即

- (1). 由基本单元构成(基本单元是不可再分的)；
- (2). 两两不交(“不重”)；
- (3). 其并为全体(“不漏”)。


 抽样单元的形成原因:

- (1). 基本抽样单元源于抽样方便且符合分析目的;
- (2). 其他抽样单元是为了特定的抽样方法和分析目的而订制的;
- (3). 抽样单元可以人为划分, 也可以是自然形成的;


例 5.1.2 继续上面的例子分析抽样框:

(1). 在粮食污染调查中, 仓库可以看成是初级单元, 而仓库中的每一包装的麻袋可视为次级单元, 甚至可以作为基本单元。

(2). 在社会经济调查中, 各级行政单位常可以作为各级抽样单元, 如做全国性调查可以将省(自治区、直辖市)一级作为一级单元; 将市、县作为二级单元; 将街道、乡、镇作为三级单元; 将居民委员会或者村民委员会作为四级单元; 而基本单元可能是住户或者个人。

 抽样框的特点:


- (1). 抽样框可以以各种形式出现: 名单、手册、地图和数据库等;
- (2). 抽样框必须是有序的, 即: 抽样单元必须编号, 且根据某种顺序排列。
- (3). 抽样框中包含的抽样单元务必要“不重不漏”, 否则将出现所谓的“抽样框误差”


 抽样框与总体的关系:

- (1). 抽样框不同于总体, 它是一个逻辑集合, 而非实体集合;
- (2). 去除逻辑映像差异, 抽样框另一个不同于总体之处在于它包含一种特殊的树状分层的抽样单元结构, 与特定的抽样方法有关;
- (3). 抽样框与总体关系密切: 抽样框是总体的一个反映, 其基本信息来自于总体。

三 抽样与样本

定义 5.5 从总体全部基本单元中抽取的部分基本单元, 称为“样本”, 每个被抽中的基本单元称为“入样单元”, 样本中所包含的抽样单元数 n 称为“样本容量”(Sample Size), 样本量 n 与总体单元总数 N 的比值 $f = \frac{n}{N}$ 称为抽样比(Sampling Fraction); 该过程称为“抽样”。

 抽样的过程决定了样本与总体之间的联系, 不同的分析目的决定不同的抽样方法。

 抽样方法是多样的:

(1). 逐个抽取与全样本抽取

样本中的 n 个单元可以逐个抽取, 也可以一次抽取 n 个单元, 后者称为“全样本方法”。

(2). 无放回抽样与有放回抽样

在逐个抽取中, 每次被抽中的单元, 即入样单元可以被放回总体中去, 也可以不放回总体中去, 前者称为有放回抽样(Sampling with replacement)或回置抽样, 后者称为无放回抽样(Sampling without replacement)或不回置抽样。

在回置抽样中，一个单元有可能被抽到两次或两次以上，故有人也称它为重复抽样；而在不回置抽样中，一个单元之多被抽到一次，不可能被重复抽样，等价于全样本抽样。


(3). 等概率抽样与不等概率抽样

抽样过程中，每个单元被抽到的概率可能相等也可能不相等，前者称为等概率抽样(Sampling with equal probabilities)，后者称为不等概率抽样(Sampling with non-equal probabilities)。


§5.1.3 总体特征与估计量

一 总体特征

定义 5.6 总体特征是指整个总体中所有单元的某种属性的概括表现或综合指标。

 总体特征与调查目标：每项调查都有特定的内容与目的，通常调查的目标量都是由总体的某些指标来表示的。

例 5.1.3 在儿童情况的调查中，全国或者某个地区的0-14岁儿童总数，儿童在总人口中所占的比例，某个年龄组的儿童的平均身高与体重，婴儿或5岁以下儿童的死亡率，用母乳喂养的婴儿在全体婴儿中的比例，学龄儿童的在校率，儿童犯有龋齿的比例等上百项指标均属于调查目标量，它们都是描述总体特征的指标。

 总体特征分类：

(1). 总体总量，或总体总和(Population Total)：

例如：0-14岁儿童总数，某市商业零售总额，某地区某年粮食产量和牲畜存栏总数等。


(2). 总体均值，或总体平均数(Population Mean)：

例如：居民平均每年用于服装消费的费用，户平均居住面积，儿童平均犯龋齿的个数等。

(3). 总体具有某种特征的个体在全体中的比例(Proportion)

例如：儿童在总人口中的比例，患某种疾病的人在总人群中的比例，即：患病率，全体选民中对某个候选人或某个提案的支持率等。这些比率常用百分率表示。

(4). 总体中两个指标的总和或平均数的比值(Ratio)

 总体特征的数量表达式：

设 \mathcal{X}, \mathcal{Y} 为总体的两项指标，总体共包含 N 个(抽样)单元，其中 N 是已知的。记指标 \mathcal{X} 在每个单元上的指标值分别为 X_1, X_2, \dots, X_N ；指标 \mathcal{Y} 在每个单元上的指标值分别为 Y_1, Y_2, \dots, Y_N ，则

(1). \mathcal{X}, \mathcal{Y} 的总体总和分别为

$$S_{\mathcal{X}} = \sum_{i=1}^N X_i$$

$$S_{\mathcal{Y}} = \sum_{i=1}^N Y_i$$

(2). \mathcal{X}, \mathcal{Y} 的总体均值分别为

$$\bar{\mathcal{X}} = \frac{1}{N} \sum_{i=1}^N X_i$$

$$\bar{\mathcal{Y}} = \frac{1}{N} \sum_{i=1}^N Y_i$$

(3). \mathcal{Y} 的是示性函数,

$$Y_i = \begin{cases} 1 & \text{若第 } i \text{ 个单元具有某个特定的特征} \\ 0 & \text{否则} \end{cases}$$

描述了单元是否具有某个特征。则该特征的总体总和为:

$$S_{\mathcal{Y}} = \sum_{i=1}^N Y_i$$

表示具有该特征的单元总数; 总体均值分别为

$$\bar{\mathcal{Y}} = \frac{1}{N} \sum_{i=1}^N Y_i$$

表示具有该特性的单元在总单元数 N 中所占的比例 P 。

(4). \mathcal{X}, \mathcal{Y} 两个指标的总和或均值比值定义为

$$R = \frac{S_{\mathcal{X}}}{S_{\mathcal{Y}}} = \frac{\bar{\mathcal{X}}}{\bar{\mathcal{Y}}}$$

二 估计量与抽样分布

定义 5.7 估计量是根据所抽取的 n 个单元的样本的样本函数; 估计量是随机变量, 其分布称为“抽样分布”。

✎ 估计量是样本函数, 样本函数是随机变量, 存在概率分布。

✎ 估计量与估计量的值

(1). 估计量

(2). 估计量的值

✎ 估计量不唯一, 需要进行评价择优。

✎ 抽样分布的获取方法:

(1). 精确(抽样)分布: 即获得统计量的精确概率分布函数。

这种情况很少, 属于经典的小样本理论。经典的情形就是正态总体下, 样本均值和方差相关的结论—Fisher定理; 另外, 在某些简单的泊松分布, 有时候也可以推出某些精确分布。

(2). 渐进分布: 指统计量当样本容量很大时的概率分布的表现;


(3). 近似分布: 指有限样本情形下, 对统计量分布的近似, 主要利用再抽样方法实现如: bootstrap方法, jackknife方法

§5.1.4 误差与精度


一 误差来源

1. 抽样误差

定义 5.8 抽样误差(Sampling Error)是由于抽样所引起的误差, 确切的说抽样误差来自于由于用样本(部分单元)估计总体(全体单元)而产生的误差。

 (概率)抽样误差的本质: 样本的随机性。

对任何一种抽样方案, 可能的样本观测值会很多, 而实际抽到的只是某一个确定的样本观测值, 从而计算获得某一个统计量值, 如果样本观测值改变, 则相应的估计量的值会不一样, 这是偶然的、随机的。

 (概率)抽样误差是可以计量与控制的。

(1). 可计量性:

抽样误差可以用各种量值表示, 这是显然的, 因为样本观测值本身就是可计量的值。

(2). 可控制性:

抽样误差可以控制在任意小的范围内, 这是因为抽样误差直接与样本量有关。

2. 非抽样误差

定义 5.9 非抽样误差(non-sampling error)不是由于抽样引起的, 它又包括调查误差, 不完整的抽样框引起的误差, 不回答误差以及由于填写或录入调查数据中的谬误而产生的误差等。


二 精度衡量


1. 点估计与均方误差

定义 5.10 设要估计的总体特征为 θ_0 , 由样本推算总体的估计量为 $\hat{\theta}$, 则

$$\text{MSE}(\hat{\theta}) = E(\hat{\theta} - \theta_0)^2$$

称为“均方误差”(MSE, Mean Square Error)。

 “均方误差”是估计量误差平方的期望, 是在 $L_2(P)$ 空间中所展开的一套估计量精度判别的经典理论。

 误差分解理论:

$$\begin{aligned}\text{MSE}(\hat{\theta}) &= E(\hat{\theta} - \theta_0)^2 \\ &= E\left[\hat{\theta} - E(\hat{\theta})\right]^2 + \left[E(\hat{\theta}) - \theta_0\right]^2 \\ &= \text{Var}(\hat{\theta}) + \text{Bias}^2(\hat{\theta})\end{aligned}$$

(1). 方差: $\text{Var}(\hat{\theta}) = E\left[\hat{\theta} - E(\hat{\theta})\right]^2$, 描述了估计量的随机波动性。

(2). 偏差: $\text{Bias}(\hat{\theta}) = E(\hat{\theta}) - \theta_0$, 描述了估计量的系统偏差。

(3). 两者不能同时达到最小, 是矛盾的; 因此, 要使精度(MSE)达到最小, 需要方差与偏差达到某种平衡点。

✎ 样本误差分解的样本版—信度与效度

设 $X = (X_1, X_2, \dots, X_n)$ 为一容量为 n 的 *i.i.d.* 样本, 且 $E(X_1) = \mu$ 。令 $S(X) = \frac{1}{n} \sum_{i=1}^n (X_i - \bar{X})^2$ 为样本方差, $\bar{X} - \mu$ 为样本偏差, 则样本均方误差有如下

$$\begin{aligned} \text{MSE}(X) &= \frac{1}{n} \sum_{i=1}^n (X_i - \mu)^2 \\ &= \frac{1}{n} \sum_{i=1}^n (X_i - \bar{X})^2 + [\bar{X} - \mu]^2 \\ &= S(X) + [\bar{X} - \mu]^2 \end{aligned}$$

(1). 信度: $S(X) = \frac{1}{n} \sum_{i=1}^n (X_i - \bar{X})^2$, 描述了在一定条件下, 多次测量时, 所得结果之间的符合程度。

(2). 效度: $\text{Bias}(X) = \bar{X} - \mu$, 描述了在一定条件下, 多次测量时, 所得结果的平均值与真实值之间的符合程度。

(3). 两者不能同时达到最小, 是矛盾的; 因此, 要使测量精度(MSE)达到最小, 需要样本方差与样本偏差达到某种平衡点。

2. 区间估计及其精度衡量

利用区间估计, 主要考虑精确度和可靠性, 前者用平均区间长度衡量, 后者用置信水平表示, 而事实上, 两者往往不能兼顾, 需要折中; 根据N-P准则, 优先考虑可靠性, 在保证一定的可靠性前提下, 尽可能保证精确度。

此处所谓的精度衡量, 指的是给定一定的置信水平, 比较区间长度的做法。

三 精度控制与样本容量(费用)

§5.1.5 抽样调查的一般流程

一 确定调研问题

调查中最重要的是明确调查目标。由此决定以下几个方面:

- (1). 调查对象—总体及其相应的指标
- (2). 分析方法—最后数据的分析方法

二 抽样方案设计

抽样方案设计主要决定于抽样框, 因为抽样框以总体及其相应指标确定为前提, 并未将来的分析方法奠定基础。抽样框决定了抽样方法, 有三个主要途径获取

- (1). 使用已有的抽样框
- (2). 补充现有的抽样框
- (3). 建立全新的抽样框

三 问卷设计

问卷设计尽管有完善的设计准则, 但是还是一门需要灵性、经验和谨慎的艺术, 其主要涉及的问题:

- (1). 提问什么问题?

- (2). 如何修辞?
- (3). 如何安排问题的顺序?

四 实施调查过程

五 数据处理分析

六 撰写调查报告

§5.2 常见抽样方法

§5.2.1 简单随机抽样

一 抽样方法

定义 5.11 (无放回)简单随机抽样(SRS: Simple Random Sampling): 从一个单元数为 N 的总体中逐个抽取单元并且无放回, 每次都在所有尚未进入样本的单元中等概率的抽取, 直到 n 个单元抽完。

(有放回)简单随机抽样(USRS: Unstricted Simple Random Sampling): 从一个单元数为 N 的总体中逐个抽取单元并且有放回, 每次都在所有单元中等概率的抽取, 直到 n 个单元抽完。

🔗 (无放回)简单随机抽样等价方法: 由于全样本方法和逐个无放回抽取是等价的, 也可以一次性从总体中抽出 n 个单元, 这也是简单随机抽样。

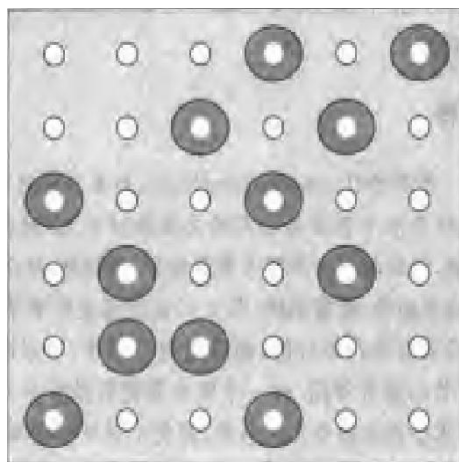


图 5.1: Simple Random Sampling

🔗 简单随机抽样的特点:

- (1). 优点: 直观;
- (2). 缺点: 一是总体规模 N 很大时抽样困难; 二是简单随机抽样中的单元都比较分散, 获取一个单元的代价很大。

二 估计量及其精度

1. 估计量

简单随机抽样中的估计方法。通常采用样本均值(平均数)作为总体均值的估计,用样本比例作为总体比例的估计,这就是简单估计。有时候为了提高精度,在有其他辅助变量存在的情况下,也可以用比估计和回归估计方法等。


2. 精度

三 样本量控制

§5.2.2 分层抽样

一 抽样方法

定义 5.12 分层抽样(Stratified Sampling): 将总体按一定的原则分成若干个子总体,每个子总体称为“层”,在每个层内进行抽样,不同层的抽样相互独立,这样的抽样称为分层抽样。如果各层内是简单随机抽样,则称为分层随机抽样。分层随机抽样的估计是先在各层内进行的,再由各层的估计量进行加权平均或求和,从而得出总体的估计量。

 分层抽样的本质: 层全部获取,不具有随机性,因而对于层的特性获取最全面的信息;层内进行随机抽样。

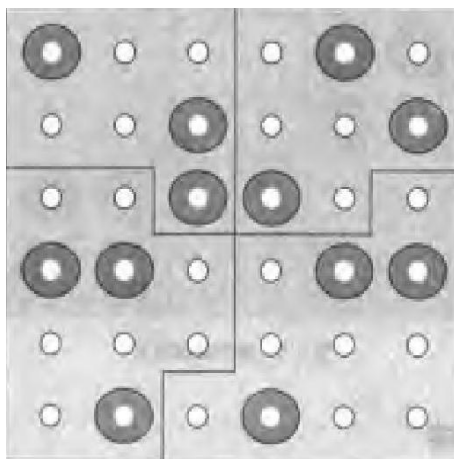



图 5.2: Stratified Sampling

 分层抽样的特点:

- (1). 分层抽样不仅适用于总体估计,同时也适用于估计各层的值。当划分的各层之间的差异较大,而层内各单元的差异较小时,分层抽样可以显著的提高估计精度。
- (2). 分层抽样的单元比较集中,有利于调查的实施。

二 估计量及其精度

1. 估计量


2. 精度

三 样本量控制

§5.2.3 整群抽样

一 抽样方法

定义 5.13 整群抽样(Cluster Sampling):如果抽样仅对初级单元进行,对抽中的初级单元调查其全部的次级抽样单元,对没有抽中的单元则不进行调查,这种抽样方法称之为“整群抽样”。所谓“群”,即指初级抽样单元。

 整群抽样的本质:初级单元随机抽样,次级单元全调查。

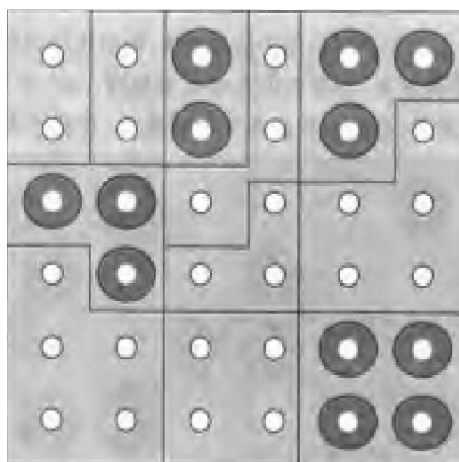



图 5.3: Cluster Sampling

 整群抽样的特点:

- (1). 缺点: 精度差
- (2). 优点: 一是样本单元比较集中, 节省费用; 二是整群抽样只需要初级单元抽样框, 简化了抽样框的编制。

二 估计量及其精度

1. 估计量
2. 精度

三 样本量控制

§5.2.4 二阶段与多阶段抽样

一 抽样方法

定义 5.14 多阶段抽样(Multi-stage Sampling): 可以看作整群抽样的发展, 在抽得初级抽样单元后, 并不调查其全部的次级单元, 而是进行再抽样, 从入选的初级单元中抽取次级单元, 这种抽样方法称之为“二阶段抽样”(Two-stage Sampling), 二阶段抽样中的第一阶段指抽取初级单元, 第二阶段是指抽取次级单元(在二阶段抽样中也就是基本抽样单元); 类似的, 可以定义三阶段抽样等。

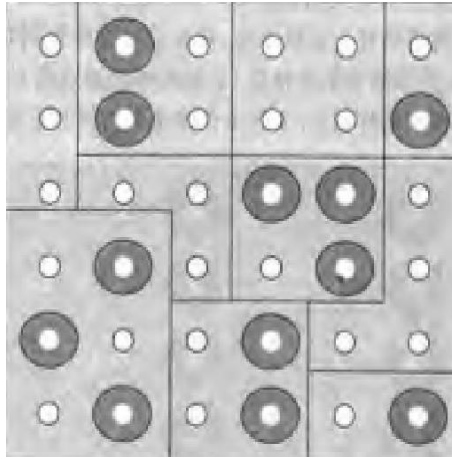


图 5.4: Muti-Stage Sampling

✎ 多阶段抽样的本质：在初级单元进行抽样，然后在获取的次级单元中继续抽样，循环下去。

✎ 多阶段抽样的特点：

- (1). 具有整群抽样的优点：抽样单元比较集中，调查方便；
- (2). 多阶段抽样中对估计量方差的估计很复杂。

二 估计量及其精度

1. 估计量
2. 精度

三 样本量控制

§5.2.5 系统抽样(等距抽样)

一 抽样方法

定义 5.15 系统抽样(Systematic Sampling): 先将总体中的抽样单元按某种顺序排列，在规定范围内随机抽取一个初级单元，然后按事先规定的规则抽取其他样本单元。特别的，如果在抽取初始单元后按相等的间距抽取其余样本单元，称为“等距抽样”。

✎ 系统抽样的本质：初始单元随机抽取，其他单元按某种规则由初始单元确定性的决定下来。

✎ 系统抽样的特点：

- (1). 优点：实施简便，只需要随机抽取初始单元；如果总体的排列具有一定的规律性，并在设定抽样规则时善加利用，则系统抽样完全可以取得很高的估计精度。
- (2). 缺点：估计精度难以确定；有的系统抽样根本就不算概率抽样。

二 估计量及其精度

1. 估计量
2. 精度

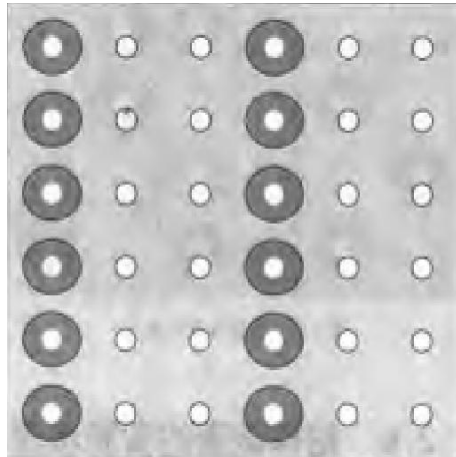


图 5.5: Systematic Sampling

三 样本量控制

§5.2.6 不等概率抽样

一 抽样方法

定义 5.16 不等概率抽样(Sampling With Unequal Probability): 在全面所述单元抽样中, 样本抽取不一定要等概率, 有时候采取不等概率抽样效果更好。

- ✎ 不等概率抽样的本质: 考虑单元本身的大小, 决定其入样的概率大小。
- ✎ 不等概率抽样的特点: 常结合其他抽样方法

二 估计量及其精度

1. 估计量
2. 精度

三 样本量控制

§5.3 SAS抽样过程——surveyselect

§5.3.1 surveyselect过程及其它随机数理论

一 过程简介

1. 功能简介
surveyselect
2. 简单随机抽样
3. 分层抽样

二 其他随机理论简介

§5.3.2 抽样方法的SAS实现

一 简单随机抽样方法

1. 无放回简单随机抽样

例 5.3.1 从1-100的整数中无放回的随机选取30个数，

(1). 不运用Surveyselect求其可能的样本。

(2). 通过Surveyselect求其可能的样本。

解： 首先我们通过不运用Surveyselect求其可能的样本。

```
data all;
  do i=1 to 100;
    output;
  end;
run;
data smp;
  set all;
  rdm=uniform(0);
run;

proc sort data=smp out=out_smp;
  by rdm;
run;

data out_smp;
  set out_smp;
  if _n_ le 30;
run;
proc sort data=out_smp;
  by i;
run;
```

接下来可以通过Surveyselect求其可能的样本。

```
proc surveyselect data=all out=b noprint
  samsize=30; /*??????*/
run;

proc surveyselect data= all out=class noprint
  samprate=0.3; /*??????*/
run;
```

接下来可以通过简单的宏变量实现：

```

%let sampsize=100;
data tmp;
  set sashelp.prdsale nobs=nobs;
  retain _cnt_ 0;
  if &sampsize > _cnt_ and ranuni(0) * (nobs + 1 - _n_) < (&sampsize - _cnt_) then
    do;
      _cnt_+1;
      output;
    end;
  drop _cnt_;
run;

```

2. 有放回简单随机抽样

例 5.3.2 从1-100中有放回的抽取30个整数，用SAS实现。

解：借助于随机数的基本思路是：利用均匀随机数，模拟1-100上的整数的等可能取值，其基本步骤是

STEP 1: 产生随机数 $rdm \sim U[0, 1]$;

STEP 2: 从1-100中等可能取出一个整数 i ;

STEP 3: 重复前两步，循环 n 次。

比较容易出现的可能解法是：

```

data all;
  do i=1 to 100;
    output;
  end;
run;
data sample_srs;
  set all;
  rdm=uniform(0);
  if (i=int(100*rdm))&(index+1 le 30) then
    do;
      output;
      index+1;
    end;
run;

proc print data=sample_srs;
run;

```

注意到上述DATA步中的内循环，知道内循环次数 $n = 100$ ，每次内循环均产生一个均匀随机数 rdm ，相应的整数随机数为 $\xi_i = [100 * rdm]$ ，则

$$\begin{aligned}
 P(A_i) &= P(\xi_i = i) = p \\
 &= \frac{1}{100}
 \end{aligned}$$

■

若令 $S = \sum_{i=1}^n \mathbf{1}_{A_i} \sim b(n, p)$, 则 $E(S) = np = 100 \times 0.01 = 1$, 则最后输出的数据记录条数平均为1条, 根本没有实现我们的想法。

正确的解法应该借助于宏编程并且使用SAS宏的data步接口进行实时数据(随机数)传递, 以产生抽样效果, 其代码为:

```
%let population=100;
data all;
  do i=1 to &population;
    output;
  end;
run;

%let sample_size=30;
data sample tem_sample_set0;
  i=100;
  delete;
run;

%macro create_uniform_sample;
  data uniform_sample;
    rdm=uniform(0);
    call symput('uniform_number',rdm);
  run;
%mend create_uniform_sample;
/*The length of data_set name must be less than 32 characters*/
%macro sample_with_replacement;
  %put &uniform_number;
  data tem_sample_set1;
    set all;
    if i=1+int(&population*&uniform_number) then output;
  run;
  data sample;
    set tem_sample_set0 tem_sample_set1;
  run;
  data tem_sample_set0;
    set sample;
  run;
%mend sample_with_replacement;
/*The basic idea can be described by the following two steps:*/
%do n=1 %to &sample_size;
  %create_uniform_sample;
  /*The first step is how to produce a uniform random number.*/
  %sample_with_replacement;
```

```

        /*Sampling with replacement*/
    %end;
%mend analyze;
%analyze;
proc print data=sample;
run;

```

二 分层抽样方法

例 5.3.3

解: libname chapt12 "f:\data_model\book_data\chapt12"; proc sql;

```

select
sex,age_cde,count(*) as cnt
from chapt12.origin
group by 1,2
;
quit;

```

```

proc sort data=chapt12.origin out=outsort;
    by sex age_cde;
run; proc surveyselect  data=outsort noprint
    method=srs rate=0.5  out=output1
    ;
    strata sex age_cde
    ;
run; proc sort data=chapt12.origin out=outsort;
    by sex age_cde;
run; proc surveyselect  data=outsort noprint method=srs
rate=chapt12.equal_sample  out=output2;
    strata sex age_cde;
run; proc surveyselect  data=outsort noprint
    method=srs rate=(0.5,0.8,0.3,0.5,0.7,0.1)  out=output3 ;
    strata sex age_cde;
run; proc surveyselect  data=outsort noprint
    method=srs rate=chapt12.unequal_sample  out=output33;
    strata sex age_cde;
run; proc surveyselect  data=outsort noprint
    method=srs rate=chapt12.unequal_sample_s1  out=output4;
    strata sex age_cde;
run; proc surveyselect  data=outsort noprint
    method=srs rate=chapt12.unequal_sample_s2  out=output5;
    strata sex age_cde;
run; proc surveyselect  data=outsort noprint

```



```

        method=srs n=chapt12.unequal_sample out=output6;
        strata sex age_cde;
run; proc surveyselect data=outsort noprint
        method=srs n=chapt12.unequal_sample_s3 out=output7;
        strata sex age_cde;
run; proc surveyselect data=outsort noprint
        method=srs n=chapt12.unequal_sample_s3 selectall out=output7;
        strata sex age_cde;
run;

```

三 宏编程

例 5.3.4

解: Libname Survey 'F:\Data_Model\Book_data\chapt12'; Options Mprint
Mlogic Mstored Sasmsstore=Survey;

```

%Macro Survey(
Input=,Method=,Outhits=,Reps=,
Strata=,Variable=,Nr=,N=,Rate=,Output=) / STORE; /* Check Input Data
Set Is Or Not To Be Defined */
%If %Sysfunc(Exist(&Input))=0 %Then %Do;
    %Put Error: &Input Data Set Do Not Exist!;
%Return;
%End;
/**Check Method,We Usually Use Srs And Urs****/
%If &Strata Eq %Then %Do;
    /**Check Sampling According N Or R***/
    %If &Nr=N %Then %Do;
        Proc Surveyselect Data=&Input Noprint
            Method=&Method &Nr=&N Out=&Output &Outhits. Reps=&Reps.
            ;
        Run;
    %End;
%Else %If &Nr=R %Then %Do;
        Proc Surveyselect Data=&Input Noprint
            Method=&Method &Nr=&Rate Out=&Output &Outhits. Reps=&Reps.
            ;
        Run;
    %End;
%Else %Do;
    %Put Error:The Value &Nr Is Only N Or R!;
%End;
%End;

```

```

%Else %If &Strata=Strata %Then %Do;
  %If &Nr=N %Then %Do;
    /***Check N Is Values Input****/
    %If "%Substr(&N,1,1)" Eq "(" %Then %Do;
    %PUT %Substr(&N,1,1);
    Proc Sort Data=&Input out=sort_&Input;
      By &Variable;
    Run;

    Proc Surveyselect Data=sort_&Input noprint
      Method=&Method N=&N Out=&Output &Outhits. Reps=&Reps.
      ;
      &Strata &Variable
      ;
    Run;
  %End;
  /***Check N Is Dataset Input****/
  %Else %Do;
    %Let Dsid=%Sysfunc(Open(&N.));
    %If &Dsid Gt 0 %Then %Do;
      %If %Sysfunc(Varnum(&Dsid,_Nsize_))<=0 %Then %Do;
        %Let Rc=%Sysfunc(Close(&Dsid));
        %Put Error: &N. Do Not Exist The Field _Nsize_!;
      %Return;
      %End;
      %Let Rc=%Sysfunc(Close(&Dsid));
    %End;
    %Else %Do;
      %Let Rc=%Sysfunc(Close(&Dsid));
      %Put Error:&N. Do Not Exist!;
    %End;
    Proc Sort Data=&Input out=sort_&Input;
      By &Variable;
    Run;

    Proc Surveyselect Data=sort_&Input noprint
      Method=&Method N=&N Out=&Output &Outhits. Reps=&Reps.
      ;
      &Strata &Variable
      ;
    Run;
  %End;

```

```

%End;
%ELSE %If &Nr=R %Then %Do;
  /***Check R Is Values Input****/
  %If "%Substr(&Rate,1,1)" Eq "(" %Then %Do;
  %PUT %Substr(&Rate,1,1);
  Proc Sort Data=&Input out=sort_&Input;
    By &Variable;
  Run;

  Proc Surveyselect Data=sort_&Input noprint
    Method=&Method Rate=&Rate Out=&Output &Outhits. Reps=&Reps.
    ;
    &Strata &Variable
    ;
  Run;
%End;
/***Check R Is Dataset Input****/
%Else %Do;
  %Let Dsid=%Sysfunc(Open(&N.));
  %If &Dsid Gt 0 %Then %Do;
    %If %Sysfunc(Varnum(&Dsid,_Rate_))<=0 %Then %Do;
      %Let Rc=%Sysfunc(Close(&Dsid));
      %Put Error: &N. Do Not Exist The Field _Rate_!;
    %Return;
  %End;
  %Let Rc=%Sysfunc(Close(&Dsid));
%End;
%Else %Do;
  %Let Rc=%Sysfunc(Close(&Dsid));
  %Put Error:&N. Do Not Exist!;
%End;
Proc Sort Data=&Input out=sort_&Input;
  By &Variable;
Run;

Proc Surveyselect Data=sort_&Input noprint
  Method=&Method Rate=&Rate Out=&Output &Outhits. Reps=&Reps.
  ;
  &Strata &Variable
  ;
Run;
%End;
%End;

```

```

        %Else %Do;
            %Put Error:The Value &Nr Is Only N Or R!;
        %End;
    %End;

    %Else %Do;
        %Put Error:The Value &Strata Is Only Strata Or Null!;
    %End;
%Mend Survey;

/*???:*/ libname survey 'f:\data_model\book_data\chapt12'; options
mprint mlogic mstored sasmstore=survey; data class1(drop=name
rename=(temp=name)); set sashelp.class; temp=substr(name,1,3); run;

data class; set sashelp.class class1; run;

data basic; set class; n=_n_; length age_cde $20; if age le 12 then
age_cde='<12'; else if 13<=age<14 then age_cde='13-14'; else
age_cde='>15'; run;

proc sort data=basic ; by sex age_cde; run;

data _null_; set basic nobs=nobs; call symput('nobs',nobs); stop;
run;
%put &nobs;

proc sql; create table work.para as select sex ,age_cde ,count(*) as
_NSIZE_ ,count(*)/&nobs as _RATE_ from basic group by 1,2 ; quit;
%Survey(Input=Basic,Method=Srs,Outhits=,Reps=1,
Strata=,Variable=,Nr=N,N=4,Rate=,Output=Test);
%Survey(Input=Basic,Method=Urs,Outhits=Outhits,Reps=2,
Strata=,Variable=,Nr=N,N=4,Rate=,Output=Test);
%Survey(Input=Basic,Method=Urs,Outhits=Outhits,Reps=1,
Strata=Strata,Variable=sex ,Nr=N,N=(4,5),Rate=,Output=Test);
%Survey(Input=Basic,Method=Urs,Outhits=Outhits,Reps=1,
Strata=Strata,Variable=sex ,Nr=R,N=,Rate=(0.4,0.5),Output=Test);
%Survey(Input=Basic,Method=Urs,Outhits=Outhits,Reps=2,
Strata=Strata,Variable=sex age_cde,Nr=N,N=para,Rate=,Output=Test);

```

第六章

简单统计分析——参数估计与假设检验

我们介绍了变量的描述性分析与相应的SAS处理。业已知道，对变量进行描述既可用一些常用统计量，又可借助统计图与统计表把现象直观地表现出来。除了对现象进行描述外，实践中我们还常常需要利用样本资料对总体进行统计推断。统计推断问题主要有两类：一是参数的估计，这已在第四章中通过MEANS过程和UNIVARIATE过程做了介绍；二是假设检验，包括参数的假设检验和非参数假设检验。另外，变量之间是否存在相关关系，也是实际统计分析中常常面临的问题之一。本章将介绍这些问题的SAS处理方法及其所使用的有关SAS过程。

§6.1 假设检验与SAS过程

§6.1.1 假设检验基础

一 问题的提出

我们先来举两个具体的例子，通过这两个例子来了解假设检验所要解决的基本问题。

例 6.1.1 为了了解城镇居民家庭消费水平是否有所提高，2002年，某市对其居民家庭进行了一次抽样调查，其中，100户被抽样家庭的调查结果列于表

表 6.1: 2002年某市居民家庭月均消费水平（抽样）

平均每户消费支出（元/月）	500	600	700	800	900	1000
家庭数	8	15	30	25	13	9

据调查，3年前该市居民家庭月均消费支出服从 $N(720, 17580)$ 分布。假定2002年城镇居民家庭月均消费支出服从正态分布，问该市居民家庭月均消费支出是否有显著提高？（显著性水平为0.05）

本例中，假定城镇居民家庭月均消费支出服从正态分布，但分布参数并未告之，需要通过检验来确定。也就是说，随机变量的概率分布是已知的，概率分布中参数却是未知的，需要通过样本数据来检验 $\mu=720$ 是否成立，这是一个典型的参数假设检验问题。

例 6.1.2 某银行正在考虑在两个相邻地区A和B之间开设一个新的营业网点。银行所关心的是这两个地区家庭平均收入是否相同。为此，在这两个地区分别抽取了20户居民家庭进行调查，调查结果列于表

问在0.05的显著性水平下，这两个地区的家庭平均收入是否有显著差异？

本例中，来自于两个地区家庭收入的数据可以看作为两个独立的样本，但是两个独立样本的总体分布并未告之。需要解决的问题是：根据抽出的两个地区家庭收入资料来

表 6.2: 地区A和B家庭平均收入水平情况

A地区家庭收入(万元)	2.5	2.9	3.2	5.3	3.8	4.2	4.0	3.9	3.3	3.1
	4.5	4.7	4.2	5.7	5.1	3.0	4.9	2.7	3.8	4.6
B地区家庭收入(万元)	3.7	4.1	4.3	3.6	3.9	3.8	4.7	4.4	5.3	5.1
	3.8	3.7	6.0	5.5	2.9	3.4	5.2	3.8	4.8	4.6

判断这两个地区的家庭平均收入是否有显著差异。更一般地说,在两个总体分布未告知的情况下,通过样本来推断两个总体的均值是否相同。这是一个典型的两样本假设检验问题,也称为两组比较问题。这类问题在现实生活中非常多。例如,为了检验广告效果,需要检验广告前后产品销售量有无显著变化?吸烟是否对健康有害?可通过从吸烟和不吸烟的人群中随机抽出若干人,构成吸烟组和不吸烟组,然后考察这两组人的某项健康指标,如肺癌患病率,以检验吸烟组的肺癌患病率是否显著高于不吸烟组等问题。

对这类问题的处理,通常有两种方法。一是两总体分布虽然未告知,但是假定它们服从某一分布,通常假定服从正态分布。在此假定下检验两总体均值是否有显著差异,这仍是参数假设检验问题。二是不假设总体服从任何分布,直接检验两总体均值是否有显著差异,这是典型的非参数假设检验问题。

二 假设检验问题

假设检验就是先对总体的参数或分布做出某种假设,然后用适当的方法根据样本对总体提供的信息,推断此假设是接受还是拒绝。该方法在现代经济学和管理学的研究中得到了广泛的应用。

定义 6.1 在两个相互对立的假设中选择用户自认为正确的假设,这类问题称为假设检验问题。

假设检验问题分类

(1). 按假设形式:

单侧检验: 在假设检验中,根据原假设和备择假设的表达形式不同可分为单侧检验和双侧检验。一般来说,单侧检验所提出的假设通常为: $: >$, $: <$ 或: $: \geq$, $: \leq$ 前者称为左侧检验,其对应的拒绝域是小于某一给定值的所有数值的集合;后者称为右侧检验,其对应的拒绝域是大于某一给定值的所有数值的集合,如例6.1的假设。

双侧检验: 所提出的假设通常为: $: =$, $: \neq$ 其对应的拒绝域是小于或大于某一给定值的所有数值的集合,如例6.2中的假设。单、双侧检验的接受域和拒绝域见图

(2). 按是否参数表达:

参数检验是指总体的分布是已知的,而其总体的分布参数却是未知的,需要根据样本数据来推断出总体的未知参数,

非参数假设检验是在总体分布未知的没有任何限定的条件下,根据样本数据推断出总体具有某种分布特征或参数间具有某种性质的假设检验

假设检验问题的构成

(1). 原假设

(2). 备择假设

✎ 假设检验的基本原理：实际推断原理

三 假设检验

四 假设检验的一般步骤

假设检验一般分为以下五个步骤：

第一步：根据研究问题的需要提出原假设和备择假设。

第二步：确定检验的统计量及其概率分布。第三步：在给定的显著性水平条件下，确定决策规则。第四步：根据样本数据计算检验统计量的值。

第五步：作出决策并加以解释。

假设检验的五个步骤可归纳为流程图。

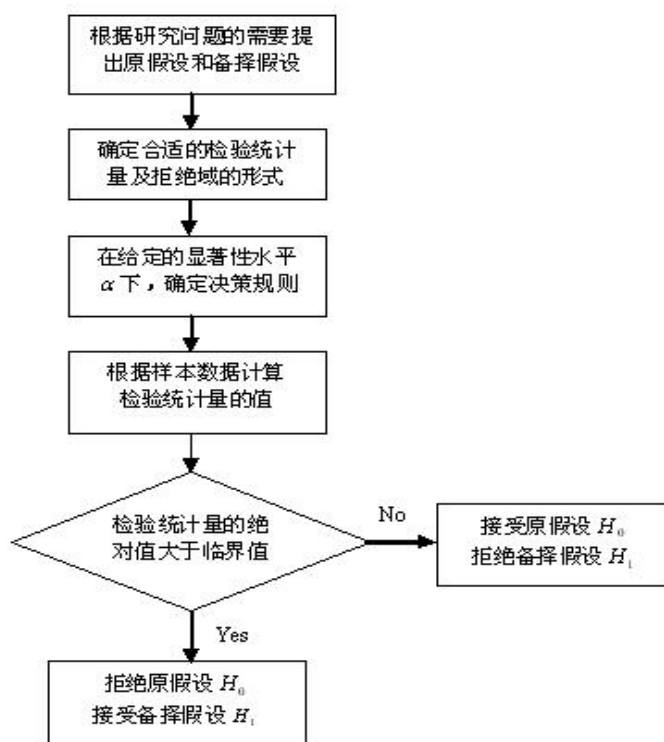


图 6.1: 假设检验的一般步骤流程图

§6.1.2 单样本检验问题

一 单样本的参数假设检验

参数的假设检验问题一般都是以正态分布为理论基础的，所以以下关于参数的假设检验问题我们都假定总体服从正态分布。

设 x_1, x_2, \dots, x_n 是来自正态总体 $N(0, \sigma^2)$ 的一个样本，其中 μ 和 σ^2 至少有一个是未知的。单样本参数假设检验通常需要检验以下两类问题：一类是根据样本观测值检验总体均值 μ 与某个给定的数 μ_0 是否有显著差异；二是根据样本观测值检验总体方差 σ^2 与某个给定的数 σ_0^2 是否有显著差异。以下我们将分别介绍这两类问题。

1. 总体均值的假设检验

STEP 1: 给定假设检验问题:

可能是双侧假设检验问题:

$$\text{原假设: } \mu = \mu_0 \iff \text{备择假设: } \mu \neq \mu_0$$

当然也可能是单侧检验问题

$$\text{原假设: } \mu = \mu_0 \iff \text{备择假设: } \mu \geq \mu_0 \text{ (右侧检验)}$$

$$\text{原假设: } \mu = \mu_0 \iff \text{备择假设: } \mu \leq \mu_0 \text{ (左侧检验)}$$

STEP 2: 确定合适的检验统计量以及拒绝域的形式:

(1). 检验统计量

当 $\sigma^2 = \sigma_0^2$ 已知时, 用 U 统计量

$$U = \frac{\bar{x} - \mu_0}{\sigma_0/\sqrt{n}} \stackrel{H_0}{\sim} N(0, 1) \quad (6.1)$$

当 σ^2 未知时, 用 t 统计量

$$T = \frac{\bar{x} - \mu_0}{S/\sqrt{n}} \stackrel{H_0}{\sim} t(n-1) \quad (6.2)$$

(2). 拒绝域的形式: 此处拒绝域形式是双侧形式 $W = \{|U| \geq c\}$

当假设检验的形式是双侧检验时, 此时的拒绝域形式当然也是双侧拒绝域; 当假设检验的形式是右侧检验时, 此时的拒绝域形式当然也是右侧拒绝域; 当假设检验的形式是左侧检验时, 此时的拒绝域形式当然也是左侧拒绝域;

STEP 3: 确定决策规则: 给定显著性水平 α , 确定拒绝域的边界—临界值(Critical Value)。

因为 $P(W) \leq \alpha$, 所以 $c \geq z_{\alpha/2}$ **STEP 4:** 给定样本观测值并计算统计量 U 的值 u :(1). $u \in W$ 或等价的 $P \text{ value} \leq \alpha \implies \text{Refuse } H_0$ (2). $u \notin W$ 或等价的 $P \text{ value} > \alpha \implies \text{Refuse } H_0$ **例 6.1.3** 例如刚才所举的“居民消费水平”的例子。**解:** SAS程序及其程序执行结果:

```
data consume;
input expend number @@;
dif=expend-720;
cards;
500 8 600 15 750 30 800 25 900 13 1000 9
;
proc means mean t prt clm alpha=0.1;
var dif;
freq number;
run;
```


程序说明：赋值语句dif=expend-720计算出每一个观测值与720之差，因此检验均值是否显著高于720就相当于检验变量dif的均值是否显著大于0。means过程要求进行假设检验，并给出变量dif的均值、t统计量的值，对应t的值和上下置信界。由于该例是在0.05水平的单侧检验，而SAS给出的都是双侧检验，由图6.1知，应取为0.1，即选项alpha=0.1。语句freq number表示每个分析变量dif的频数是number，因为本例中的数据是一个经初步整理后的分组资料。程序运行结果如下(略)：

从以上结果可以看出，检验变量dif的t值为3.1675056，检验概率Prob|t|>T—T—的值为0.0020，小于给定的显著性水平0.05，从而在0.05的水平上可推断出变量dif的均值显著大于0（从下置信界为19.98也可看出这一点），也即居民家庭月均消费支出显著大于720元。从上面输出结果还可以得到2002年该市居民家庭月均消费支出的90%的置信区间为（739.98元，784.02元）。

2. 总体方差的假设检验

STEP 1: 给定假设检验问题：

可能是双侧假设检验问题：

$$\text{原假设: } \sigma^2 = \sigma_0^2 \iff \text{备择假设: } \sigma^2 \neq \sigma_0^2$$

当然也可能是单侧检验问题

$$\text{原假设: } \sigma^2 = \sigma_0^2 \iff \text{备择假设: } \sigma^2 \geq \sigma_0^2 \text{ (右侧检验)}$$

$$\text{原假设: } \sigma^2 = \sigma_0^2 \iff \text{备择假设: } \sigma^2 \leq \sigma_0^2 \text{ (左侧检验)}$$

STEP 2: 确定合适的检验统计量以及拒绝域的形式：

(1). 检验统计量

当 $\mu = \mu_0$ 已知时，用 χ^2 统计量

$$\chi^2 = \frac{\sum_{i=1}^n (x_i - \mu_0)^2}{\sigma_0^2} \stackrel{H_0}{\sim} \chi^2(n) \quad (6.3)$$

当 μ 未知时，用 χ^2 统计量

$$\chi^2 = \frac{\sum_{i=1}^n (x_i - \bar{x})^2}{\sigma_0^2} \stackrel{H_0}{\sim} \chi^2(n-1) \quad (6.4)$$

(2). 拒绝域的形式：此处拒绝域形式是双侧形式 $W = \{|\chi^2| \leq c_1\} \cup \{|\chi^2| \geq c_2\}$

当假设检验的形式是双侧检验时，此时的拒绝域形式当然也是双侧拒绝域；当假设检验的形式是右侧检验时，此时的拒绝域形式当然也是右侧拒绝域；当假设检验的形式是左侧检验时，此时的拒绝域形式当然也是左侧拒绝域；

STEP 3: 确定决策规则：给定显著性水平 α ，确定拒绝域的边界—临界值(Critical Value)。

因为 $P(W) \leq \alpha$ ，所以 $c_1 = \chi_{1-\alpha/2}^2(n-1)$, $c_2 = \chi_{\alpha/2}^2(n-1)$

STEP 4: 给定样本观测值并计算统计量 χ^2 的值 u ：

(1). $u \in W$ 或等价的 $P \text{ value} \leq \alpha \implies \text{Refuse } H_0$

(2). $u \notin W$ 或等价的 $P \text{ value} > \alpha \implies \text{Refuse } H_0$

例 6.1.4 例如刚才所举的“居民消费水平”的例子。已经知道三年前居民家庭月均消费支出的方差为17580，现在需要检验三年后居民家庭月均消费支出的方差有否变化，即是否仍为17580。

解： 根据题意可令：

$$\text{原假设: } \sigma^2 = 17580 \iff \text{备择假设: } \sigma^2 \neq 17580$$

需要检验两者中哪一个成立。SAS程序及其程序执行结果：

```
data consume;
input expend number @@;
cards;
500 8 600 15 750 30 800 25 900 13 1000 9
;

proc means var;
var expend;
freq number;
output out=test var=varex;
run;

data a(drop=_type_);
set test;
k=_freq_ -1;
chisq=k*varex/17580;
p=1 probchi (chisq,k);
ci1=cinv (0.025,k);
ci2=cinv (0.975,k);

proc print data=a noobs;
run;
```

程序说明：程序第一部分创建了一个名为consume的SAS数据集；第二部分调用means过程计算变量expend的方差，并用output语句把输出结果存放到数据集test中，计算过的方差用varex命名；第三部分用set语句又创建了一个SAS数据集a，其中变量_type_和_freq_是由output语句产生的，分别表示类型和观测个数，变量k定义的 χ^2 自由度，变量chisq计算 χ^2 统计量值，变量p利用概率函数probchi(chisq,k)计算大于统计量值chisq的概率，而变量ci1, ci2则利用 χ^2 分布的分位数函数cinv分别计算0.05和0.975的自由度为k的 χ^2 临界值；最后利用print过程把数据集a打印出来。程序运行结果如下(略)：

以上检验结果：CI1;CHISQ;CI2和p=0.48081均表明：在0.05的显著性水平下，居民家庭月均消费支出的方差没有发生显著性变化

另外，由以上输出结果不难计算出 σ^2 的置信区间为：(13553.75, 23726.47)。

二 单样本的非参数假设检验

在单样本的参数检验中，我们假定总体服从正态分布，但实际情况是否如此，需要通过样本来检验。当检验结果认为总体是服从正态分布时，利用前面介绍的参数检验方法是可行的，但当检验结果认为总体不服从正态分布时用前面的检验方法就不合理了，此时需要用非参数检验。由于参数检验的效率要优于非参数检验，所以通常要先对总体分布进行检验，然后再决定使用参数检验还是非参数检验。为此我们先介绍总体分布的拟合优度检验，它本身也是一种非参数检验。

1. 总体分布的拟合优度检验

拟合优度检验是根据样本的经验分布对总体分布做出的估计。因此，当总体确实服从某一假定的分布，即假设为真时，其样本观测频数应与由总体算出的理论频数大致相等。这里之所以说是大致相等，是因为即使假设为真，由于抽样存在误差，这两者数值仍有可能不会完全相等。问题是这种差别大到什么程度，能否用抽样误差来解释。如果这种差别可以归结为抽样误差，我们就没有理由拒绝对总体分布所做的假设；反之，如果这种差别不能完全归结为抽样误差，我们就有理由怀疑假设的合理性。

为了更好地理解这一点，我们先来看一个简单的例子。

例 6.1.5 某企业欲了解其产品订单的分布情况，在随机选择的一周中发现，其订单频数分布如表所示

表 6.3: 订单频数分布表

星期一	星期二	星期三	星期四	星期五	合计
7	12	15	11	15	60

问：该企业的订单在每星期的5天中是均匀分布的吗？取显著性水平 $\alpha=0.05$ 。

解： 如果订单分布是均匀的，则每天收到订单的概率应该相等，即都等于0.2。因此，一星期60份订单中，从理论上说每天应收到12份，但实际情况并不是这样，见表。为了概括它们之间的差别，记观测频数为 f_{oi} ，理论频数为 f_{ei} ，其中 $i = 1, 2, \dots, k$ (k 表示所分的组数，如本例中 $k=5$)，K. Person提出并证明了以下统计量：

$$\chi^2 = \sum_{i=1}^k \frac{(f_{oi} - f_{ei})^2}{f_{ei}} \stackrel{H_0}{\sim} \chi^2(k-1) \quad (6.5)$$

本例中有一个约束：

$$\sum_{i=1}^k f_{oi} = \sum_{i=1}^k f_{ei} = 60 \quad (6.6)$$

所以本例自由度为 4。

在显著性水平 $\alpha=0.05$ 的条件下，查得 $\chi_{0.05}^2(4)=9.488$ 。由于 $\chi^2 = 3.66 \leq 9.488$ ，所以我们接受订单在每星期的5天中是均匀分布这一假设。

本例用SAS程序实现如下：

```
data chisq;input foi fei@@;
dif= (foi fei) ;div=dif*dif/fei;
```

```

cards;
7    12    12    12    15    12    11    12    15    12
;
proc means sum; var div;
output out=test sum=chisq;run;
data a;set test;k=_freq_ 1;
p=1 probchi (chisq,k) ;ci1=cinv (0.025,k) ;
ci2=cinv (0.975,k) ;
proc print data=a noobs;run;

```

下面我们再来看一个正态性检验的例子。正态性假设是经典统计分析最常见的假设，因而经常需要进行检验，SAS为其提供了非常方便的检验方式，至少可以有以下三种方式：

- (1). 形如一般形式，如上例进行计算；
- (2). 利用univariate过程；
- (3). 利用Insight菜单系统。

例 6.1.6 下列程序中的数据来自于2001年我国内地31个省、自治区及直辖市城镇居民家庭人均收入资料，试检验这批数据是否来自正态分布，也即城镇居民家庭人均收入是否服从正态分布？

解： 关于正态性检验，在第5章已有介绍。以下就是利用第5章介绍的UNIVARIATE过程给出的SAS程序，读者可以运行这段程序以检验数据的正态性。

本例用SAS程序实现如下：

```

data income;
input x@@;
cards;
10349.69 8140.50 5661.16 4724.11 5129.05 5357.79 4810.00
4912.88 11718.01 6800.23 9279.16 5293.55 7432.26 5103.58
6489.97 4766.26 5524.54 6218.73 9761.51 5834.43 5358.32
6275.98 5894.27 5122.21 6324.64 7426.32 5124.24 4916.25
5169.96 4912.40 5644.86
;proc univariate normal;
var x;run;

```

利用Insight系统求解略去。

2. 总体均值的非参数检验

我们知道，在大样本情况下，即使总体不服从正态分布，样本均值仍近似服从正态分布。因此，在大样本情况下仍可利用MEANS过程中的T检验对总体均值进行检验。但在小样本的情况下，当总体不服从正态分布时，样本均值就不能使用正态分布

进行近似，因此也不能利用MEANS过程中的T检验对总体均值进行检验，此时需要利用UNIVARIATE过程的符号检验或威尔克森秩和检验。

注意，在UNIVARIATE过程中的符号检验或威尔克森秩和检验检验的是总体均值是否为0，若要检验总体均值是否为某一给定值，常对观测数据进行的变换，再检验变量Y的均值是否为0。

§6.1.3 两样本检验问题

在现实生活中，我们经常会遇到这样一类检验问题：男性职工是否比女性职工的工资高？广告是否有助于提高产品的销售量，即广告前后产品的销售量是否有显著差别？某项技术革新是否带来了产量的增加或成本的节约、产品质量的提高？配股或增发前后股票收益率是否有显著变化？非PT类上市公司的资产负债率与PT类公司的资产负债率是否有显著差别，等等。这些问题都与两组之间的比较有关，其中有些是独立的，即两组中的样本来自于两个独立总体，称为独立组。如男女职工工资高低问题中的男女职工总体、PT和非PT类公司资产负债率高低问题中的PT类公司与非PT类公司总体都可看作两个独立总体；有些是成对的，即两组中的样本来自于一个总体的不同时间或不同处理下的观测，称为成对组。如广告前后产品的销售量就是来自于不同时间下某个企业销售量的数据，某种产品采用某种新包装和不采用新包装的销售量数据就是来自于不同处理的数据，这类数据都可看作是成对数据。在两样本比较中，把数据分成两组的变量称为分类变量，如性别、上市公司类型（PT类和非PT类）、是否广告等。对于独立组和成对组，由于它们的假设条件不同，所采用的检验统计量不同，所以在SAS系统中检验这两类问题的SAS过程也往往不同，下面我们将分别予以介绍。

一 两独立组的假设检验

在进行两个独立组的假设检验时，检验假设通常是：

$$H_0 : \mu_A = \mu_B, \text{ 即两独立组的均值相等}$$

$$H_0 : \mu_A \neq \mu_B, \text{ 即两独立组的均值不等}$$

其中 μ_A, μ_B 分别为 A, B 两组的均值。

为了给出以上假设的检验统计量，需检验两样本资料是否满足以下两个条件：

- (1) 正态性，即各组资料是否来自于独立的正态分布；
- (2) 方差齐次性，即两总体的方差是否相等。

当两样本满足 (1)、(2) 两个条件时，可以采用一般的参数T检验，其检验统计量为：

$$t = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{s^2(\frac{1}{n_1} + \frac{1}{n_2})}} \sim \chi^2(n_1 + n_2 - 2) \quad (6.7)$$

其中 $s^2 = [(n_1 - 1)s_1^2 + (n_2 - 1)s_2^2] / (n_1 + n_2 - 2)$ 。

当两样本满足条件 (1) 而不满足条件 (2) 时，在大样本情况下可采用参数的近似正态分布进行检验，在小样本情况下，可采用参数的近似T检验或非参数的威尔克森秩和检验。

参数的近似T检验统计量为：

$$t' = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}} \sim \chi^2(n_1 + n_2 - 2) \quad (6.8)$$

当两个条件都不满足时，采用非参数的威尔克森秩和检验，其检验统计量参见Univariate过程。

例 6.1.7 某银行正在考虑在两个相邻地区A和B之间开设一个新的营业网点。银行所关心的是这两个地区家庭平均收入是否相同。为此，在这两个地区分别抽取了20户居民家庭进行调查，调查结果列于表

问在0.05的显著性水平下，这两个地区的家庭平均收入是否有显著差异？

解： 本例用SAS程序实现如下：

```
data income;
input area$ income@@;
cards;
A 2.5   B 3.7   A 3.2   B 4.3   A 3.8   B 3.9   A 4.0   B 4.7   A 3.3   B 5.3
A 4.5   B 3.8   A 4.2   B 6.0   A 5.1   B 2.9   A 4.9   B 5.2   A 3.8   B 4.8
A 2.9   B 4.1   A 5.3   B 3.6   A 4.2   B 3.8   A 3.9   B 4.4   A 3.1   B 5.1
A 4.7   B 3.7   A 5.7   B 5.5   A 3.0   B 3.4   A 2.7   B 3.8   A 4.6   B 4.6
;
proc sort; by area;run;
proc univariate normal;
var income;by area;run;
proc ttest ; class area;
var income;run;
```

前四行是检验观测数据是否服从正态分布的。从中可以看出，A地区家庭收入数据的正态性检验的W值为0.970438，其对应的p值为0.7546，说明该地区家庭收入服从正态分布。类似可得到B地区家庭收入也服从正态分布（见输出第四行）。由此得到样本数据的正态性得到满足，可采用T检验或近似T检验，此时对应的检验过程名为ttest。

在进行两独立组均值的比较时，首先要看输出6.4最后一行，此行标志为For H0: Variances are equal（两组方差相等），它给出了两总体的方差是否相等的检验结果：检验的F值由 $F = 1.31$ ，对应的p值由 $\text{Prob}_i F = 0.5658$ 给出，由此可得出两总体方差没有显著性差异，此时可以用精确T检验对两组的均值是否相等进行检验，这可从标有Variances的一列中找到标有Equal（方差相等）的行，然后检查t值及其对应的概率。本例中t值为1.3228，对应的p值即 $\text{Prob}_i T$ 的概率为0.1938，大于0.05，由此可得出结论：两地区家庭收入没有显著性差异。

注意若两总体的方差检验结果为不相等，则要看Unequal所在行的t值和对应的概率。

例 6.1.8 下列程序中的数据来自2002年上市公司年报，其中，变量type中的pt表示该公司为ST或PT类，而nopt则表示该公司为非ST或非PT类公司，变量rate代表公司的资产负债率。试根据给定的资料分析ST或PT类公司的平均资产负债率是否显著高于非ST或PT类公司。显著性水平取为0.05。

解： 根据题目要求，首先检验两组数据是否来自正态总体，这由程序中的univariate过程来解决。经检验发现非ST或非PT类公司的资产负债率数据服从正态分布，而ST或PT类

则不服从正态分布，因此平均资产负债率是否有显著性差异的检验应采用非参数的威尔克森秩和检验。非参数检验的过程名为`npar1way`，程序中该过程后的选项`wilcoxon`表示要给出威尔克森秩和检验，`class`语句把观测按变量`type`分成两类，而`var`语句则要求对变量`rate`进行均值比较，程序及其运行结果如下：

```
data zichfz;
input type$ rate@@;
cards;
pt    99.4    pt    94.8    pt    38.4    pt    52.7    pt    92.1
pt    87.9    pt    334.2  pt    86.9    pt    134.5  pt    74.9
pt    69.9    pt    48.0    pt    104.9  pt    67.8    pt    60.8
pt    59.5    pt    62.0    pt    75.4    pt    715.2  pt    15.3
pt    224.6  pt    90.6    pt    86.7    pt    65.4    pt    77.1
pt    354.2  pt    59.7
nopt   31.3    nopt   54.7    nopt   29.7    nopt   40.0    nopt   55.1
nopt   32.6    nopt   59.2    nopt   46.9    nopt   52.9    nopt   29.1
nopt   64.8    nopt   35.0    nopt   56.6    nopt   44.5    nopt   52.3
nopt   21.8    nopt   52.0    nopt   28.0    nopt   24.0    nopt   13.5
nopt   29.8    nopt   67.1    nopt   17.1    nopt   48.1    nopt   30.8
nopt   32.6    nopt   24.1
;
proc sort; by type;run;
proc univariate normal;
var rate;by type;run;
proc npar1way wilcoxon;
class type;var rate;run;
```

输出的前半部分是来自于两类公司资产负债率数据的正态性检验结果。从中可以看出，来自于非ST或非PT类公司的资产负债率检验的W值为0.950262，其对应的概率值 $Pr_i W$ 为0.2376，大于0.05，说明该组数据服从正态分布；来自于ST或PT类公司的资产负债率检验的W值为0.56449，其对应的概率值 $Pr_i W$ 为0.0001，小于0.05，说明该组数据不服从正态分布，此时应采用非参数检验。

输出的后半部分是来自于两类公司资产负债率数据非参数的wilcoxon秩和检验结果。一般地，解释wilcoxon秩和检验的输出结果主要看“ $Prob_i -Z-$ ”后面的p值大小（见输出6.5的倒数第四行）。本例中 $p=0.0001 < 0.05$ ，结合非ST或非PT类公司的Mean Score为16.4074074，而ST或PT类公司的Mean Score为38.5925926，所以可以得出结论：ST或PT类公司的资产负债率显著高于非ST或非PT类公司的资产负债率。输出6.5中的最后三行分别给出了近似T检验和Kruskal-Wallis近似卡方检验结果，这些结果与wilcoxon秩和检验的结果是一致的，即ST或PT类公司的资产负债率显著高于非ST或非PT类公司的资产负债率。

二 成对组的假设检验

在进行两个独立组的假设检验时，检验假设通常是：

$H_0: \mu_D = 0$, 即两独立组的均值差为零

$H_0: \mu_D \neq 0$, 即两独立组的均值差非零

其中 μ_D 分别为 A, B 两组的均值之差。

以上假设的检验方法和检验统计量随样本差值数据是否来自正态总体而定。当差值数据来自正态总体时, 把差值看作为一个新变量, 运用MEANS过程中的参数T检验; 当差值不是来自正态总体而是任意其他分布时, 则需要运用UNIVARIATE过程中的符号检验或威尔克森符号秩检验。独立组和成对组两样本比较检验总结列于表

表 6.4: 两样本比较的检验统计量

检验类型	独立组	成对组
参数检验	两样本的T检验	成对组的T检验
非参数检验	wilcoxon秩和检验	wilcoxon符号秩检验

例 6.1.9 某企业准备更新其产品包装, 为此进行了一次实验调查。它们随机地选择了15个在地理位置、销售条件等各方面都差不多的商店, 然后用新包装代替原来的老包装进行销售。一个月后, 这15家商店的月销售量和采用新包装前一个月的月销售量情况如下

问: 新包装对产品的销售量有无显著影响? 显著性水平为0.05。

解: 根据题目要求, 程序编辑如下:

```
data package;
input sale1 sale2@@;
dif=sale2-sale1;
cards;
66 72 70 75 75 68 79 87 65 84 90 73 85 70 82 83
97 95 95 90 92 82 73 78 71 69 69 74 77 86
;proc univariate normal;
var dif; run;
```

程序说明: dif语句计算新老包装的销售量之差, 选用单变量过程univariate, 并对差值数据进行正态性检验。程序运行结果如

输出给出了新包装对产品销售量有无显著性影响的检验结果。一般来说, 利用univariate过程检验差值数据平均水平与0是否有显著性差异, 首先要看差值数据是否服从正态分布。如果服从正态分布, 则可利用参数T检验, 否则应进行非参数检验。本例中, 差值的正态性检验W值为0.969102, 其对应的概率 $Pr_i W$ 为0.8072 (见输出的最后一行), 远大于0.05, 说明差值数据服从正态分布, 此时利用参数的T检验可行。由输出6.6的第一行可以看到, t值为0, 对应的概率 $Pr_i T$ 为1.0000, 说明差值的均值与0没有显著性差异, 从而得到新包装对产品平均销售量没有显著性影响。输出6.6还给出了符号检验和符号秩检验, 这些检验的结果与T检验的结果是一致的, 也强烈支持新包装对产品平均销售量没有显著性影响的结论。

例 6.1.10 春兰股份（代号600854）于2001年7月24日向外公告：它将于2001年8月14日增发6000万股。为考察公告发出前后一段时间股票收益率有无显著变化，现收集了该公司自2001年7月9日到2001年8月13日共22个交易日的收盘价资料（见以下程序），问增发公告发出前后股票收益率是否有显著差异？显著性水平为0.1。

解： 根据题目要求，先创建一个名为stock的SAS数据集，其中变量price2，price1分别为公告前后股票的日收盘价，变量rate1，rate2分别为按以下公式计算的日收益率：

$$r = \frac{p_{t+1} - p_t}{p_t} \times 100\% \quad (6.9)$$

变量dif计算公告前后股票收益率之差。程序最后调用univariate过程以比较公告前后的收益率，选项normal要求对变量rate1，rate2，dif分别做正态性检验。本例SAS程序及其运行结果如下：

```
data stock(keep=rate1 rate2 dif);
input price1 price2@@;
rate1=(price1-lag(price1))/lag(price1)*100;
rate2=(price2-lag(price2))/lag(price2)*100;
dif=rate2-rate1;
cards;
25.79 24.90 24.37 24.88 23.24 25.20 22.11 25.10 22.00 24.99 22.12
24.78 20.29 24.90 19.70 24.80 20.46 26.05 19.98 25.41 20.68 25.46
;

proc univariate normal;
var rate1 rate2 dif;
run;
```

输出表明，公告前11天股票平均收益率为0.24%，且股票收益率不服从正态分布，这由结果“W: Normal 0.815161 PrjW 0.0220”得到证明。而公告后11天股票平均收益率为2.1%，且股票收益率服从正态分布，这由结果“W: Normal 0.959251 PrjW 0.7642”得到证明。公告前后收益率之差的数据也服从正态分布，这可由结果“W: Normal 0.977532 PrjW 0.9455”得到证明，由此可运用T检验来检验公告前后收益率是否有显著性差异。由输出“T: Mean=0 1.984328 Prj—T— 0.078”可知，假设均值为0的T检验值为1.984328，对应的概率为0.078；0.1，说明至少有90%的把握，收益率差值的平均值与0有显著性差异。由于收益率差值的平均值Mean=2.351617，所以可以得出结论：公告前的收益率要显著高于公告后的收益率，实际上公告后的平均收益率是负值，换句话说，股市不看好春兰股份的增发。

§6.2 相关分析与CORR过程

第七章

线性回归分析

§7.1 统计方法与实例分析

§7.1.1 多元线性回归分析基础

一 变量间的两类关系及其研究方法

1. 变量及变量关系

我们经常与变量打交道，譬如正方形的边长 a 与面积 S 都是变量，电路中的电阻 R 、电压 V 、电流强度 I 等也都是变量，对一个人来讲，年龄、性别、身高、体重、血压等也都是变量，它们分别是处在一个“共同体”(系统)中的若干个变量。变量间常见的关系有两类：

(1) 确定性关系：

譬如正方形的面积与边长之间有关系： $S = a^2$ ，电路中有欧姆定律 $V = IR$ 等。

这些变量间的关系完全是已知的，可以用函数 $y = f(x)$ 来表示， x （可以是向量）给定后， y 的值就唯一确定了。

(2) 相关关系：

变量间有关系，但是不能用函数来表示，譬如：

例 7.1.1 人的身高 x 与体重 y ，两者间有关系，一般来讲，身高较高的人体重也较重，但是同样身高的人的体重可以是不同的。医学上就利用这两个变量有关系，给出了一些经验公式来确定一个人是否过于“肥胖”或“瘦小”。

例 7.1.2 人的脚掌的长度 x 与身高 y ，两者间有关系，一般来讲，脚掌较长的人身高也较高，但是同样脚掌长度的人的身高可以是不同的。公安机关在破案时，常常根据案犯留下的脚印来推测罪犯的身高。

例 7.1.3 钢的强度 y 与钢水中碳、锰、... 许多元素的含量有关，但是成分相同的钢的强度可以是不完全相同的。在炼钢厂，可以通过钢水中的化学成分来预测钢的强度，或通过控制钢水中的化学成分来得到符合要求强度的钢，这在质量管理中是非常有用的。

这类变量间有关系，但是不能用确切的函数形式来表示，对于给定的 x ， y 可以不同，因此 y 是随机变量，它有一个条件分布，其分布与 x 有关，称这类变量间的关系为不确定关系或者相关关系。既然它们间有关系，那么就需要寻找它们间定量关系的表达式，这就是回归分析的任务。

不过需要注意，有时这两类关系会转化。随着研究的深入，有些相关关系会转化成函数关系；而有些变量间是有函数关系，但是在测量过程中由于误差的存在，表现出来的可能是相关关系。

2. 不确定性关系的三种研究方法

3. 不确定性关系与回归分析

回归分析便是研究变量间相关关系的一门学科。它通过对客观事物中变量的大量观察或试验获得的数据，去寻找隐藏在数据背后的相关关系，给出它们的表达形式——回归函数的估计。

y 与 x 有关，称 x 为自变量(预报变量)， y 为因变量(响应变量)， x 给定后， y 有一个分布，我们关心的是 y 的均值，它是 x 的函数：

$$f(x) = E(Y|X = x) = \int_{-\infty}^{\infty} yp(y|x)dy.$$

这便是 y 关于 x 的回归函数——条件期望，也就是我们要寻找的相关关系的表达式。在本课程的研究中，我们需要作一些约束：

✎ y 是随机变量，给定 x 后， y 的分布是什么？不同的假定会产生不同的回归模型，我们只讨论 y 服从正态分布的情况(条件分布是正态的，不一定要求绝对分布)。但是，对于自变量为随机变量的情形，必须要求其联合分布为正态的，当然要求边缘分布(绝对分布)也是正态的；

✎ y 可以是一维的，也可以是多维的。我们只讨论 y 一维的情况；

✎ x 可以是一维的，也可以是多维的，可以是随机变量，也可以是一般变量。我们只讨论 x 是非随机变量的情况；

✎ $f(x)$ 可以是线性的，也可以是非线性的。我们主要讨论线性的情况，也涉及一些非线性的。

以下各章主要讨论的内容是：

(1) 在 x 为一般变量， y 服从正态分布，其期望是 x 的线性函数 $f(x) = \beta_0 + x'\beta$ ，利用所收集的数据，研究参数估计及有关假设的检验；

(2) 数据是否符合模型给出的假定？需要进行诊断，诊断当违背假定时，要给出补救措施；

(3) 应用。

二 线性模型

在实际中影响随机变量 y 取值的自变量可能不止一个，设有 t 个： x_1, x_2, \dots, x_t ，为此需要建立 y 与 x_1, x_2, \dots, x_t 间的相关关系 ($t \geq 2$)。譬如：化工产品的得率 y 与温度 x_1 、压力 x_2 、配比 x_3 ，... 等有关。在研究 y 与 x_1, x_2, \dots, x_t 间的关系时就不象一元那么直观，无法借用图象的帮助。此时，常常由经验或直接假定 y 与 x_1, x_2, \dots, x_t 间为线性相关的关系。

一般的，当获得 $(x_1, \dots, x_t, y)'$ 的 n 个独立观测 $(x_{i1}, \dots, x_{it}, y_i), i = 1, 2, \dots, n$ 后，假设其服从以下模型：

$$\begin{cases} y_i = \beta_0 + \beta_1 x_{i1} + \dots + \beta_t x_{it} + \varepsilon_i, & i = 1, 2, \dots, n \\ \varepsilon_i \text{ 独立同分布, 且 } \varepsilon_1 \sim N(0, \sigma^2) \end{cases} \quad (7.1)$$

其中 $y_i, i = 1, \dots, n$ 为反应变量, 一般假设为单变量连续型随机变量; $(x_{i1}, \dots, x_{it})' i = 1, \dots, n$ 为非随机变量, 是可以实际观测或者严格控制的; $\beta_0, \beta_1, \dots, \beta_t$ 是回归系数, 它们是未知参数; $\varepsilon_i, i = 1, \dots, n$ 为随机误差, 是不可观测的。称模型 (7.1) 为线性模型, 其中 $E(y|x) = \beta_0 + \beta_1 x_1 + \dots + \beta_t x_t$ 为多元回归函数。

为了数学描述和推理方便, 常常将模型 (7.1) 写成矩阵形式。先引入几个矩阵记号:

$$Y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}, X = \begin{pmatrix} 1, x_{11}, x_{12}, \dots, x_{1t} \\ 1, x_{21}, x_{22}, \dots, x_{2t} \\ \vdots, \vdots, \vdots, \ddots, \vdots \\ 1, x_{n1}, x_{n2}, \dots, x_{nt} \end{pmatrix}, \beta = \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_t \end{pmatrix}, \varepsilon = \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{pmatrix}$$

则模型 (7.1) 可以等价的写为

$$\begin{cases} Y = X\beta + \varepsilon, \\ \varepsilon \sim N(0, \sigma^2 I_n) \end{cases} \quad (7.2)$$

其中 Y 为 n 维观测向量; X 为 $n \times (t+1)$ 数据矩阵, 称为结构矩阵; β 为回归系数向量; ε 为 n 维误差向量。

在模型 (7.1) 或者 (7.2) 中, 蕴含以下几个基本假设:

假设 7.1 “线性”假设: y 关于 x 的回归函数, 它是 x 的线性函数, 即 $E(y|x) = \beta_0 + \beta_1 x_1 + \dots + \beta_t x_t$

此时, 我们称 $E(y|x)$ 是 $t+1$ 维线性空间中的一个超平面, 这种假定是合理的。实际上 $E(y|x) = f(x_1, x_2, \dots, x_t)$ 可能是一个曲面, 一般是光滑的, 在某点的一个小领域中可以用多项式去逼近, 通过变换, 就成为线性的, 因此这一假定具有普遍性。事实上, 严格而言, 这是一种局部线性逼近的思想, 它在现代非参数统计学中占据极为重要的地位, 特别包括核方法(局部常数光滑)、局部线性光滑以及局部多项式光滑均是以此一思想为基础, 从而演化出种类繁多的统计方法。

假设 7.2 “齐方差”假设: 观测值或者误差的条件方差不依赖于 x , 且保持为常数, 即 $D(\varepsilon_i) = \sigma^2$ 。

假设 7.3 “独立性”假设: 各观察值独立, 即 y_1, y_2, \dots, y_n 相互独立。

“独立性”假设常常可以减弱为“线性无关”假设, 即: $\text{Cov}(\varepsilon_i, \varepsilon_j) = 0, i \neq j$, 此时结合假设 7.2 两者合称为 Gauss-Markov 假设(常简称为 G-M 条件), 此时模型变成经典的 G-M 模型:

$$\begin{cases} Y = X\beta + \varepsilon, \\ E(\varepsilon) = 0, D(\varepsilon) = \sigma^2 I_n \end{cases} \quad (7.3)$$

假设 7.4 “正态性”假设: Y 是 n 维正态的。

正态性假设一般被认为属于比较强的假定, 但是在此条件下常见统计量均有精致的小样本理论, 主要是有精确抽样分布; 然而, 根据中心极限定理(CLT), 如果 G-M 条件满足, 可以证明常见统计量均有类似的大样本理论, 主要是有极限分布。所以 G-M 条件

除了揭示了最小二乘估计重要的小样本性质(BLUE)之外,从抽样分布角度可以归入“正态”这一类(在数理统计上,其严格说法为:分布的“吸收域”为正态分布)。

类似于第一章,本章的任务包括:(1).从数据出发估计 $\beta_0, \beta_1, \dots, \beta_t$,并研究其性质;(2).对方程、系数作检验—这在多元回归中十分重要;(3).应用—预测。

三 最小二乘法

类似于一元线性回归,我们通过要求偏差平方和达到最小获得最小二乘估计。若记

$$Q(\beta_0, \beta_1, \dots, \beta_t) = \sum_{i=1}^n (y_i - \beta_0 - \beta_1 x_{i1} - \dots - \beta_t x_{it})^2 \quad (7.4)$$

那么最小二乘估计 $\hat{\beta}_0, \hat{\beta}_1, \dots, \hat{\beta}_t$ 满足

$$Q(\hat{\beta}_0, \hat{\beta}_1, \dots, \hat{\beta}_t) = \min_{\beta_0, \beta_1, \dots, \beta_t} Q(\beta_0, \beta_1, \dots, \beta_t) \quad (7.5)$$

由于 Q 关于 $\beta_0, \beta_1, \dots, \beta_t$ 的为二次型且其偏导数存在,故LSE应满足如下方程组:

$$\begin{cases} \frac{\partial Q}{\partial \beta_0} = 0 \\ \frac{\partial Q}{\partial \beta_j} = 0, \quad j = 1, 2, \dots, t \end{cases} \quad (7.6)$$

整理后得

$$\begin{cases} \sum_{i=1}^n (y_i - \beta_0 - \beta_1 x_{i1} - \dots - \beta_t x_{it}) = 0 \\ \sum_{i=1}^n (y_i - \beta_0 - \beta_1 x_{i1} - \dots - \beta_t x_{it}) x_{ij} = 0, \quad j = 1, 2, \dots, t \end{cases} \quad (7.7)$$

称此方程组为正规方程组。由其第一式可知: $\beta_0 = \bar{y} - \beta_1 \bar{x}_1 - \dots - \beta_t \bar{x}_t$,将它代入方程组的后面各式,可得到正规方程组的另一形式:

$$\begin{cases} l_{11}\beta_1 + l_{12}\beta_2 + \dots + l_{1t}\beta_t = l_{1y} \\ l_{21}\beta_1 + l_{22}\beta_2 + \dots + l_{2t}\beta_t = l_{2y} \\ \dots \\ l_{t1}\beta_1 + l_{t2}\beta_2 + \dots + l_{tt}\beta_t = l_{ty} \end{cases}$$

其中

$$\begin{aligned} l_{iy} &= \sum_{k=1}^n (x_{kj} - \bar{x}_i)(y_k - \bar{y}) = \sum_{k=1}^n (x_{kj} - \bar{x}_i)y_k \\ &= \sum_{k=1}^n x_{ki}y_k - \frac{1}{n} \left(\sum_{k=1}^n x_{ki} \right) \left(\sum_{k=1}^n y_k \right), \quad i = 1, 2, \dots, t \\ l_{ij} &= \sum_{k=1}^n (x_{ki} - \bar{x}_i)(x_{kj} - \bar{x}_j) \\ &= \sum_{k=1}^n x_{ki}x_{kj} - \frac{1}{n} \left(\sum_{k=1}^n x_{ki} \right) \left(\sum_{k=1}^n x_{kj} \right), \quad i, j = 1, 2, \dots, t \end{aligned} \quad (7.8)$$

从中解得 $\hat{\beta}_1, \hat{\beta}_2, \dots, \hat{\beta}_t$, 则 $\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x}_1 - \dots - \hat{\beta}_t \bar{x}_t$ 。

回归方程的表示也有两种形式:

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \cdots + \hat{\beta}_t x_t = \bar{y} + \hat{\beta}_1(x_1 - \bar{x}_1) + \cdots + \hat{\beta}_t(x_t - \bar{x}_t)$$

从中可知, 超平面必过 $(0, 0, \cdots, 0, \hat{\beta}_0)$ 与 $(\bar{x}_1, \bar{x}_2, \cdots, \bar{x}_t, \bar{y})$ 。

若引入矩阵符号, 则

$$\begin{aligned} Q(\beta) &= \|Y - X\beta\|^2 \\ &= Y'Y - 2Y'X\beta + \beta'X'X\beta \end{aligned} \quad (7.9)$$

其中 $\beta'X'Y = (Y'X\beta)' = Y'X\beta$ 。

回归系数的最小二乘估计需满足的正规方程组为:

$$\begin{aligned} \frac{\partial Q(\beta)}{\partial \beta} &= -2X'Y + 2X'X\beta \\ &= -2X'(Y - X\beta) = 0 \end{aligned} \quad (7.10)$$

化简为:

$$\begin{aligned} \sum_{i=1}^n x_i(y_i - x_i'\beta) &= 0 \\ X'(Y - X\beta) &= 0 \end{aligned} \quad (7.11)$$

移项整理可得

$$\begin{aligned} \left(\sum_{i=1}^n x_i x_i' \right) \beta &= \sum_{i=1}^n x_i y_i \\ X'X\beta &= X'Y \end{aligned} \quad (7.12)$$

当 $(X'X)^{-1}$ 存在时, β 最小二乘估计为

$$\begin{aligned} \hat{\beta} &= \left(\sum_{i=1}^n x_i x_i' \right)^{-1} \sum_{i=1}^n x_i y_i \\ &= (X'X)^{-1} X'Y \end{aligned} \quad (7.13)$$

今后称 $A = X'X$ 为正规方程组的系数矩阵, $B = X'Y$ 为正规方程组的常数项向量, $C = (X'X)^{-1}$ 为相关矩阵。并且总假设 C 存在, 或者等价的

假设 7.5 “可识别”假设: X 是列满秩的, 即, $\text{Rank}(X)=t+1$ 。

该条件在文献中常被称为“可识别”条件(Identification Conditions), 从理论上而言就是要求给定观测数据 $(x_i, y_i), i = 1, \cdots, n$ 条件下, 回归系数能够被唯一的估计出来, 此时又称回归系数是“可估的”(Estimable)。注意到要求结构矩阵列满秩暗含样本观测量 $n \geq t + 1$, 关于这点今后不再重申。

用矩阵表示十分简洁, 求 $\hat{\beta}$ 的步骤如下: ①写出 X, Y ; ②求出 $A = X'X$, $B = X'Y$ 与 $C = A^{-1}$; ③ $\hat{\beta} = CB$, 其中关键是写出合适的 X, Y 。

下面从多个角度解释最小二乘法的本质。

注 7.1.1 “最小二乘法”的几何解释—最小二乘法与正交投影

(1).

(2).

注 7.1.2 “最小二乘法”与最大似然法—最小二乘法的“拟似然”本质

(1).

(2).

注 7.1.3 “最小二乘法”与矩估计方法

(1).

(2).

例 7.1.4 用矩阵形式写出一元回归模型

$$\begin{cases} y_i = \beta_0 + \beta_1 x_i + \varepsilon_i, i = 1, 2, \cdots n \\ \varepsilon_i \text{i.i.d.} \sim N(0, \sigma^2) \end{cases} \quad (7.14)$$

并用矩阵形式求出 β_0, β_1 的最小二乘估计。

解： 记

$$Y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}, X = \begin{pmatrix} 1, & x_1 \\ 1, & x_2 \\ \vdots & \vdots \\ 1, & x_n \end{pmatrix}, \beta = \begin{pmatrix} \beta_0 \\ \beta_1 \end{pmatrix}, \varepsilon = \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{pmatrix}$$

则模型 (7.14) 可以表示成

$$\begin{cases} Y = X\beta + \varepsilon, \\ \varepsilon \sim N(0, \sigma^2 I_n) \end{cases} \quad (7.15)$$

进一步写出 A, B, C

$$A = X'X = \begin{pmatrix} n & \sum x_i \\ \sum x_i & \sum x_i^2 \end{pmatrix}, B = X'Y = \begin{pmatrix} \sum y_i \\ \sum x_i y_i \end{pmatrix}$$

$$C = A^{-1} = \frac{1}{n(\sum x_i^2 - n\bar{x}^2)} \begin{pmatrix} \sum x_i^2 & -n\bar{x} \\ -n\bar{x} & n \end{pmatrix} = \frac{1}{nl_{xx}} \begin{pmatrix} \sum x_i^2 & -n\bar{x} \\ -n\bar{x} & n \end{pmatrix}$$

则 β 的最小二乘估计为

$$\hat{\beta} = CB = \frac{1}{nl_{xx}} \begin{pmatrix} \sum x_i^2 & -n\bar{x} \\ -n\bar{x} & n \end{pmatrix} \cdot \begin{pmatrix} \sum y_i \\ \sum x_i y_i \end{pmatrix} = \begin{pmatrix} \bar{y} - \hat{\beta}_1 \bar{x} \\ l_{xy}/l_{xx} \end{pmatrix} \quad \blacksquare$$

例 7.1.5 写出下列 t 元中心化模型

$$\begin{cases} y_i = \beta'_0 + \beta_1(x_{i1} - \bar{x}_1) + \cdots + \beta_t(x_{it} - \bar{x}_t) + \varepsilon_i, i = 1, 2, \cdots, n \\ \varepsilon_i \text{独立同分布, 且} \varepsilon_1 \sim N(0, \sigma^2) \end{cases} \quad (7.16)$$

中的 X, Y, A, B, C , 并求出 $\beta'_0, \beta_1, \cdots, \beta_t$ 的最小二乘估计。

解: 记

$$Y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}, X = \begin{pmatrix} 1, (x_{11} - \bar{x}_1), \cdots, (x_{1t} - \bar{x}_t) \\ 1, (x_{21} - \bar{x}_1), \cdots, (x_{2t} - \bar{x}_t) \\ \vdots \quad \vdots \quad \ddots \quad \vdots \\ 1, (x_{n1} - \bar{x}_1), \cdots, (x_{nt} - \bar{x}_t) \end{pmatrix}, \beta = \begin{pmatrix} \beta'_0 \\ \beta_1 \\ \vdots \\ \beta_t \end{pmatrix}$$

由此可得 A, B

$$A = X'X = \begin{pmatrix} n & 0 & \cdots & 0 \\ 0 & l_{11} & l_{11} & l_{1t} \\ \cdots & \cdots & \ddots & \cdots \\ 0 & l_{t1} & l_{1t} & l_{tt} \end{pmatrix} = \begin{pmatrix} n & 0 \\ 0 & L \end{pmatrix}, B = X'Y = \begin{pmatrix} n\bar{y} \\ l_{1y} \\ \cdots \\ l_{ty} \end{pmatrix} = \begin{pmatrix} \bar{y} \\ l \end{pmatrix}$$

其中 l_{ij}, l_{iy} 定义同公式 (7.8) 且 $L = (l_{ij})_{t \times t}, l = (l_{1y}, l_{2y}, \cdots, l_{ty})'$ 。进一步若假设 L^{-1} 存在, 记为 $L^{-1} = (l^{ij})_{t \times t}$, 则

$$C = A^{-1} = \begin{pmatrix} 1/n & 0 \\ 0 & L^{-1} \end{pmatrix}$$

则回归系数的最小二乘估计为

$$\hat{\beta} = CB = \begin{pmatrix} 1/n & 0 \\ 0 & L^{-1} \end{pmatrix} \cdot \begin{pmatrix} \bar{y} \\ l \end{pmatrix} = \begin{pmatrix} \bar{y} \\ L^{-1}l \end{pmatrix} \quad \blacksquare$$

关于例 7.1.5 告诉我们一些很有用的启示, 特别包括两点:

注 7.1.4 “中心化”模型优点明显, 应用广泛。特别是在求逆时降低一阶, 节省计算量。因此在实际计算中常常采用中心化模型。但是当原有的数据已经具有性质 $\sum_k x_{ki} = 0, \sum_k x_{ki}x_{kj} = 0, i \neq j$ 时, 或者讲, A 已经是对角阵时, 不需要再中心化了。

注 7.1.5 “中心化模型”的前提是带常数项的线性模型 (7.1), 否则就构成一个参数约束

$$\beta'_0 - \left(\sum_{i=1}^t \beta_i \bar{x}_i \right) = 0 \quad (7.17)$$

其最小二乘解形式会不一样。如果带有常数项 β_0 , 则其最小二乘估计 $\hat{\beta}_0$ 与 $\hat{\beta}'_0$ 的关系为:

$$\hat{\beta}'_0 - \left(\sum_{i=1}^t \hat{\beta}_i \bar{x}_i \right) = \hat{\beta}_0 \quad (7.18)$$

一般而言, 利用中心化模型给出具体数据要进行计算时, 通常用数值表达式为方便。步骤如下:

- 1). 求出各 $\sum_k x_{ki}, i = 1, 2, \cdots, t, \sum_k y_i$;
- 2). 求出 $\sum_k x_{ki}^2, \sum_k x_{ki}x_{kj}, \sum_k x_{ki}y_k, i, j = 1, 2, \cdots, t, \sum_k y_k^2$;

- 3). 求出各 $l_{ij}, l_{iy}, i, j = 1, 2, \dots, t$;
 4). 解正规方程组得到 $\hat{\beta}_1, \hat{\beta}_2, \dots, \hat{\beta}_t$, 再求 $\hat{\beta}_0$ 。

下面看一个具体的数值计算的例子。

例 7.1.6 某市连续12年主要百货商店营业额 y 与该市在业人口总收入 x_1 , 当年竣工住宅面积 x_2 有关, 数据见表 7.1, 试建立 y 关于 x_1, x_2 的回归方程。

表 7.1: 百货商店营业额及其影响要素

序号	x_1 (万元)	x_2 (万平方米)	y(万元)
1	76428	9.0	8239.3
2	77932	7.8	8332.4
3	80243	5.5	8647.9
4	82975	5.0	8974.4
5	95247	10.8	9365.3
6	88159	3.5	9393.6
7	116206	6.2	12208.8
8	128950	10.8	13666.7
9	147495	18.4	15521.3
10	183172	15.7	18297.6
11	210317	32.5	23260.2
12	248521	45.5	27327.5

解: 建立中心化线性模型, 根据上述一般算法, 计算正规方程的系数阵及常向量, 从中解出 $\hat{\beta}_1, \hat{\beta}_2$, 并利用公式 (7.18) 计算 $\hat{\beta}_0$ 。

由表 7.1 中数据, 可求得

$$\begin{aligned}
 \sum x_{i1} &= 1535645, & \bar{x}_1 &= 127970.4166, \\
 \sum x_{i2} &= 170.7, & \bar{x}_2 &= 14.225, \\
 \sum y_i &= 163235, & \bar{y} &= 13602.9166, \\
 \sum x_{i1}^2 &= 2.33517047247 \times 10^{11}, & l_{11} &= 3.69999167449167 \times 10^{10}, \\
 \sum x_{i2}^2 &= 4192.61, & l_{22} &= 1764.4025, \\
 \sum x_{i1}x_{i2} &= 29335010.8, & l_{12} &= 7490460.675, \\
 \sum x_{i1}y_i &= 2.49432572278 \times 10^{10}, & l_{1y} &= 4054006263.21667, \\
 \sum x_{i2}y_i &= 3161121.62, & l_{2y} &= 839103.745000001, \\
 \sum y_i^2 &= 2667965115.14, & l_{yy} &= 447493013.056667.
 \end{aligned} \tag{7.19}$$

(其中 l_{yy} 是后面检验用的), 求 $\hat{\beta}_1, \hat{\beta}_2$ 的正规方程组为

$$\begin{cases} (3.69999 \times 10^{10})\hat{\beta}_1 + (7490461) \times \hat{\beta}_2 = 4054006263.21667, \\ (7490461) \times \hat{\beta}_1 + 1764 \times \hat{\beta}_2 = 839103.745000001. \end{cases}$$

解得其最小二乘估计为

$$\hat{\beta}_1 = 0.0946, \quad \hat{\beta}_2 = 74.1514.$$

常数项估计为

$$\begin{aligned} \hat{\beta}_0 &= \bar{y} - \hat{\beta}_1 \bar{x}_1 - \hat{\beta}_2 \bar{x}_2, \\ &= 13602.9166 - 0.0946 \times 127970.4166 - 74.1514 \times 14.225, \\ &= 442.11152464. \end{aligned}$$

从而所求得的 y 关于 x_1, x_2 的回归方程为

$$\hat{y} = 442.11152464 + 0.0946x_1 + 74.1514x_2. \quad \blacksquare$$

例 7.1.7 称量设计, 用天平称物体通常带有一定的误差, 为提高称量的精度常要将一个物体称若干次, 再取其平均。若要同时称量几个物体, 那么可以作为回归问题安排一个适当的称量方案, 以便在不增加称量总次数的情况下增加每一物体称量的次数, 以提高称量的精度。现设有四个物体 A, B, C, D 其重量分别为 $\beta_1, \beta_2, \beta_3, \beta_4$, 按以下方案称重:

1). 把四个物体都放在天平右盘, 左盘放上砝码, 使天平达到平衡, 记砝码重为 y_1 , 则

$$y_1 = \beta_1 + \beta_2 + \beta_3 + \beta_4 + \varepsilon_1$$

2). 把 A, B 两个物体都放在天平右盘, 左盘放上 C, D , 为使天平达到平衡, 要放上砝码 y_2 , 若砝码放左盘, 则记 $y_1 > 0$, 放右盘, 则记 $y_1 < 0$, 则

$$y_2 = \beta_1 + \beta_2 - \beta_3 - \beta_4 + \varepsilon_2$$

3). 把 A, C 两个物体都放在天平右盘, 左盘放上 B, D , 为使天平达到平衡, 要放上砝码 y_3 , 符号同 2), 则

$$y_3 = \beta_1 - \beta_2 + \beta_3 - \beta_4 + \varepsilon_3$$

4). 把 A, D 两个物体都放在天平右盘, 左盘放上 B, C , 为使天平达到平衡, 要放上砝码 y_4 , 符号同 2), 则

$$y_4 = \beta_1 - \beta_2 - \beta_3 + \beta_4 + \varepsilon_4$$

上述各次测量都会产生误差, $\varepsilon_1, \varepsilon_2, \varepsilon_3, \varepsilon_4$ 分别表示称量时产生的随机误差, 试计算各物体的重量。

解： 记

$$Y = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{pmatrix}, X = \begin{pmatrix} 1, & 1, & 1, & 1 \\ 1, & 1, & -1, & -1 \\ 1, & -1, & 1, & -1 \\ 1, & -1, & -1, & 1 \end{pmatrix}, \beta = \begin{pmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \\ \beta_4 \end{pmatrix}$$

由此可得 A, B

$$A = X'X = \begin{pmatrix} 4 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 \\ 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 4 \end{pmatrix} = 4I_4, B = X'Y = \begin{pmatrix} y_1 + y_2 + y_3 + y_4 \\ y_1 + y_2 - y_3 - y_4 \\ y_1 - y_2 + y_3 - y_4 \\ y_1 - y_2 - y_3 + y_4 \end{pmatrix}$$

由于 A 是对角阵，所以 $C = A^{-1}$ 很容易求得

$$C = A^{-1} = \frac{1}{4}I_4$$

从而 β 的最小二乘估计为

$$\hat{\beta} = CB = \frac{1}{4} \begin{pmatrix} y_1 + y_2 + y_3 + y_4 \\ y_1 + y_2 - y_3 - y_4 \\ y_1 - y_2 + y_3 - y_4 \\ y_1 - y_2 - y_3 + y_4 \end{pmatrix}$$

这个例子启发我们，若能使 $A = X'X$ 化为对角阵就能使计算工作量大为简化。在一定条件下这是可以办到的，后面第四章将会讨论其中的一种情况，有时为做到这一点，还得对如何收集数据加以合理安排，这是“试验设计”一科中的内容。

四 线性假设的检验问题

五 统计诊断理论

六 变量选择与压缩估计方法

七 M估计方法

M估计方法是与对应的统计模型紧紧联系在一起的，与计量经济学中著名的广义矩估计方法(GMM)关系密切。

M估计产生的根源，就现在看来，至少有三个相互联系的方面：

1). 数据具有“不一致性”，一种情形是含有少量被污染的数据，称为“异常值”(Outliers)。需要回归对应少量极端值不敏感的特征，如：中位数，于是产生了中位数回归以及对应LADE方法，以至于一般的统计模型和一般的M方法。

2). 用户关心的特征不一样，有的场合用户直接关心的并非均值，而是其他特征，这是直接来自于模型本身的推动力；

3). 经济学意义，如经济学中的正交性条件，导致了类似于GMM方法

M估计方法突破最小二乘法的约束，同时也就意味着均值模型被其他特征模型所突破——估计方法和统计模型是紧紧联系在一起的。

八 结构突变与非线性回归

有时候觉得线性模型就好像“皇帝的新装”，人们稍微留心一下就会很快会觉得这个模型虽看似完美，然而对其实际效能却是大大的有疑问：在实际生活中，到底有少关系会如此简单，以至于能用线性模型来描述呢？怀疑是显然的，可是我们不知道被谁给“忽悠住了”？也不知道从何时开始不敢出声了？也许其创始大牛们，诸如：数学王子高斯，大数学家和哲学家拉普拉斯、马尔科夫等等几百年来树起的权威吓住我们了；也许是线性模型的理论太完美了，以至于纯粹数学家们不愿意再离开这片净土，去追寻真实的世界；更也许，不明所以的普通人们只是跟着风似地崇拜起来了一线性模型不知不觉被抬上神坛，被祭奠起来。任何事物若取得了无限信任，那将都是危险的，现世的东西都必将随风逝去，何者能被永驻，更别谈小小的人类智慧？高斯、马尔科夫等前辈先贤们若地下有灵，看到此情此景后不知道会如何自处？也许会惊叹唏嘘、啼笑皆非吧？

“门限自回归模型”(TAR)的创立者、著名统计学家汤家豪教授(Tong, 2005)曾说：“事实证明，过度沉迷于线性模型的研究，推迟了非线性模型的诞生，这是时间序列的不幸，也是统计学的不幸。”毫无疑问，在某些统计领域，如：实验设计等，线性模型由于其本身的合理性——特别是其内在蕴含的“可加性”(Additivity)——而依然占据中心地位，但是我想，是时候打破线性模型的神话让其从神坛走向平凡了，是时候对线性假设进行大胆怀疑甚至某些场合应该毫不犹豫进行否定的时候了，至少，作为年轻一代的统计学人不能再像吸食鸦片一般迷醉其中——顶多她只是一件经聪明绝顶的数学家们之手被精雕细琢之后的珍贵艺术品，可以细细把玩、慢慢欣赏，但是绝不要把她当成这个世界模型的唯一、更万万不能把她就误会成了这个真实的世界。

如果把“线性性”理解成真实世界的规律在局部时空下的一种近似，则如果涉及到更大范围的规律，就有可能明显“偏离”(Bias)这种“近似线性”(Approximate Linearity)，建模这种“偏离”的一种自然选择就是“突变分析”和“门限模型”，或者说分段建模的思想。以此为基础，也许分段的规律性正是更大范围内统一规律的体现，此时若有稳固的专业背景支持则可以提出其他合理的非线性参数模型，如果没有专业背景支持，则可以建立非参数模型或半参数模型，引入“算法建模”的思想，一般而言，有比较系统的理论来描述非线性世界，大体分为三种：

§7.1.2 过程介绍

- 一 Reg过程
- 二 GLM过程
- 三 Mix过程

§7.2 程序实现方法

§7.2.1 基本多元回归分析

下面以例 ?? 数据演示多元回归分析的软件实现方法。

1. 创建数据集与查看数据集

下面的SAS代码展现了一个典型的SAS程序，从这一段代码可以看到SAS程序的基本面貌，同时也能初步领略其语句风格。

SAS源码 7.1: Create a dataset

```
1 data huishou;
2   input x1 x2 y;
3   cards;
4   76428 9.0      8239.3
5   77932 7.8      8332.4
6   80243 5.5      8647.9
7   82975 5.0      8974.4
8   95247 10.8     9365.3
9   88159 3.5      9393.6
10  116206 6.2     12208.8
11  128950 10.8    13666.7
12  147495 18.4    15521.3
13  183172 15.7    18297.6
14  210317 32.5    23260.2
15  248521 45.5    27327.5
16 ; run;
17
18 proc print data=huishou; run;
```

语句说明：

(1). SAS程序一般由两部分构成：数据步和过程步。数据步为一组命令，第一个语句为data语句(形如第一行所示)，须以关键字 DATA 开始，一般以run语句结尾；而过程步也是一组命令，第一个语句为proc语句(形如第十四行所示)，须以关键字 Proc 开始，一般也以run语句结尾。

(2). 上述每一行均为一个SAS命令。SAS命令语句以关键字开始，以分号“；”来标志句子的结束，其中语汇之间至少空一格。

(3). “data huishou;”

该命令建立一个名字为“huishou”的临时数据集。该数据集将会保存在一个名为“work”的数据库中，只是占据一定内存，而并未存储在硬盘等物理介质上，一旦退出SAS系统，系统释放相应内存空间，该数据集将不再存在。

(4). “input x1 x2 y;”

定义两个变量，均为数值型。

(5). “cards;”

告诉SAS系统下面将是数据行。该语句后面是数据行，数据行的数据与input中的变量顺序保持一致。数据输入完成后，需要另起一行输入命令结束符号“；”，标志数据行的结束。

(6). “run;”

这是数据步或者过程步的最后一条语句，告诉SAS系统，该步结束。

(7). “proc print data=huishou;”

该语句标志一个过程步的开始，同时调用print过程，指定数据集为huishou，否则默认为当前数据集。该过程主要为了查看数据集，体现了SAS程序的一种常见风格，也是

我们避免出错的一种好习惯，即：每次输入数据集后，可以此过程输出数据集确认是否出错。

2. 画散点图

有了数据之后，通常先做描述性分析，特别的，经常画出各变量之间的散点图，以直观观测其关系，为后面的理论模型假设奠定基础。

SAS源码 7.2: 散点图

```
1 proc plot data=huishou;  
2   plot y*x1=*;  
3   title Scatter Graph between y and x1;  
4 run;
```

(1). “proc plot data=huishou;”

该语句标志plot过程步开始，指明数据集为 huishou。

(2). “plot y*x1=*;”

该语句画变量 y (纵轴变量)与 x1 (横轴变量)的散点图。

(3). “title 'Scatter Graph between y and x1';”

指明散点图的名字为Scatter Graph between y and x1。

3. 建立回归方程及显著性检验

SAS源码 7.3: Create a dataset

```
1 proc reg data=huishou ; /* graphics; */  
2   model y=x1 x2;  
3   title Regression Models between y and x1 x2;  
4 run;
```

(1). “proc reg data=huishou/method=none; ”

该语句标志reg过程步开始，指明数据集为 huishou。

(2). “model y=x1 x2;”

指定模型，方法为一般回归，“/method=none”可省略。如果希望计算过有点的线性回归方程，则可以用选项“/noint”。以输入结果 为例，一般包含以下几部分结果，

① 回归模型的名字信息。

② 方差分析。主要是方差分析表，还包括一些简单的拟合优度指标

③ 参数估计。包括参数的最小二乘估计值，标准误以及系数显著性检验的T检验统计量的值及p值。

(3). “title 'Regression Models between y and x1 x2';”

指明回归分析结果文字的名字为Regression Models between y and x1 x2。

4. 预测

有了回归方程，且评价较好后，可以用于讨论预测问题。只需在上述代码后面加入如下语句

SAS源码 7.4: Create a dataset

```
1  print  cli;  /* print clm; */
2  run;
```

选项 cli 表示要求计算，在第 i 个观测点 x_i 重新试验所得 y_i' 的预测值和 95% 预测区间；选项 clm 表示要求计算均值 $E(y_i)$ 的点估计值及 95% 置信限。还可以在 REG 过程中用 plot 语句要求画出置信限的图形

SAS源码 7.5: Create a dataset

```
1  plot y*x1=o pred.*x1=-
2      l95.*x1=L u95.*x1=U/overlay;
3  run;
```

§7.2.2 基本诊断分析

1. 线性性诊断

下面以例 ?? 数据多元回归模型中如何诊断线性性的软件实现方法。

下面的SAS代码展现了线性性的残差诊断方法，包括以下几种残差图： $\hat{y} - e$ ， $x - e$ ， $\hat{y} - r$ ， $x - r$ 。

SAS源码 7.6: Create a dataset

```
1  data speed_power;
2  input x y @@;
3  cards;
4      22.0    64.03    18.0    52.90
5      20.0    62.47    16.0    48.84
6      18.0    54.94    14.0    42.74
7      16.0    48.84    12.0    36.63
8      14.0    43.73    10.5    32.05
9      12.0    37.48    13.0    39.68
10     15.0    46.85    15.0    45.79
11     17.0    51.17    17.0    51.17
12     19.0    58.00    19.0    56.65
13     21.0    63.21    21.0    62.61
14     22.0    64.03    23.0    65.31
15     20.0    59.63    24.0    63.89
16 ; run;
17
18 proc print data=speed_power; run;
19
20 proc reg data=speed_power;
21     model y=x/r;
22     plot (R.)*x;
23     plot (student.)*x;
```

```

24      plot (student.)*p.;
25      title  Regression Models between y and x;
26  run;

```

语句说明：

(1). 该段程序实现了回归方程进行一元线性拟合,并计算其预测值、残差、标准化残差,并画出相应的残差图。

(2). “model y=x/r;”

该语句建立 y 与 x 之间的简单回归模型;选项 $/r$ 计算出回归方程的预测值、残差和标准化残差及其它一些相关量,计算结果保存在上表第四五列;此处也可用 $/p$,但只计算预测值和残差;此处也可用 $/cli$ 计算出预测值和残差和预测值的区间估计。

(3). “plot (R.)*x;”

该命令画残差 e 与 x 的残差图。

(4). “plot (student.)*x;”

该命令画学生化残差 r 与 x 的残差图。

(5). “plot (student.)*p.;”

该命令画学生化残差 r 与预测值 \hat{y} 的残差图。

从上述几个残差图经常可以诊断出线性性是否成立?但是受限于直观!如果观测值有重复,则可以使用失拟检验方法。下面的程序段可以实现该检验。

SAS源码 7.7: 线性性的失拟检验方法

```

1  proc glm data=speed_power;
2      plot y*x1=*;
3      title  Scatter Graph between y and x1;
4  run;

```

注意到由于REG过程不能对失拟作检验,这里利用GLM过程间接实现具有重复试验数据回归分析。其分析结果为(后面补上)

上面的表格中的过程是GLM中得到的结果,最后 1 行是对失拟检验的结果, $F = 0.17$, $P = 0.9122$,说明SSL基本上是由试验误差等偶然因素引起的,故需将失拟部分合并到误差中去,再检验回归方程是否显著。注意:第 1 部分实际已给出将失拟部分合并到误差中去作检验的结果了。

2. 误差的独立性诊断

有时候齐方差条件不一定满足,需要诊断分析。常用的方法失 D.W 方法。下面以例 ?? 中数据描述回归模型中如何诊断独立性的软件实现方法。

SAS源码 7.8: 独立性诊断

```

1  data speed_power;
2      input x y @@;
3      cards;
4      22.0    64.03    18.0    52.90
5      20.0    62.47    16.0    48.84

```



```

6      18.0    54.94    14.0    42.74
7      16.0    48.84    12.0    36.63
8      14.0    43.73    10.5    32.05
9      12.0    37.48    13.0    39.68
10     15.0    46.85    15.0    45.79
11     17.0    51.17    17.0    51.17
12     19.0    58.00    19.0    56.65
13     21.0    63.21    21.0    62.61
14     22.0    64.03    23.0    65.31
15     20.0    59.63    24.0    63.89
16 ; run;
17
18 proc print data=speed_power; run;
19
20 proc reg data=speed_power;
21     model y=x/r;
22     plot (R.)*x;
23     plot (student.)*x;
24     plot (student.)*p.;
25     title Regression Models between y and x;
26 run;

```

语句说明：

(1). “proc plot data=huishou;”

该语句标志plot过程步开始，指明数据集为 huishou。

(2). “plot y*x1='*';”

该语句画变量 y (纵轴变量)与 x1 (横轴变量)的散点图。

(3). “title 'Scatter Graph between y and x1';”

指明散点图的名字为Scatter Graph between y and x1。

3. 正态性检验

SAS源码 7.9: Create a dataset

```

1  proc reg data=huishou ; /* graphics; */
2      model y=x1 x2;
3      title Regression Models between y and x1 x2;
4  run;

```

(1). “proc reg data=huishou/method=none; ”

该语句标志reg过程步开始，指明数据集为 huishou。

(2). “model y=x1 x2;”

指定模型，方法为一般回归，“/method=none”可省略。如果希望计算过有点的线性回归方程，则可以用选项“/noint”。以输入结果 为例，一般包含以下几部分结果，

① 回归模型的名字信息。

② 方差分析。主要是方差分析表，还包括一些简单的拟合优度指标

③ 参数估计。包括参数的最小二乘估计值，标准误以及系数显著性检验的T检验统计量的值及p值。

(3). “title 'Regression Models between y and x1 x2';”

指明回归分析结果文字的名字为Regression Models between y and x1 x2。

§7.2.3 变量选择

下面以例 ?? 血红蛋白与微量元素数据演示多元回归分析中变量选择的软件实现方法。

1. 创建数据集与查看数据集

下面的SAS代码展现了一个典型的SAS程序，从这一段代码可以看到SAS程序的基本面貌，同时也能初步领略其语句风格。

SAS源码 7.10: Create a dataset

```

1  data hemo;
2    input x1 x2 x3 x4 y;
3    cards;
4  54.89   30.86   448.70  1.010   13.50 72.49   42.61   467.30  1.640
5  13.00 53.81   52.86   425.61  1.220   13.75 64.74   39.18   469.80
6  1.220   14.00 58.80   37.67   456.55  1.010   14.25 43.67   26.18
7  395.78  0.594   12.75 54.89   30.86   448.70  1.010   12.50 86.12
8  43.79   440.13  1.770   12.25 60.35   38.20   394.40  1.140   12.00
9  54.04   34.33   405.60  1.300   11.75 61.23   37.35   446.00  1.380
10 11.50 60.17   33.67   383.20  0.914   11.25 69.69   40.01   416.70
11 1.350   11.00 72.28   40.12   430.80  1.200   10.75 55.13   33.02
12 445.80  0.918   10.50 70.08   36.81   409.80  1.190   10.25 63.05
13 35.07   384.10  0.853   10.00 48.75   30.53   342.90  0.924   9.75
14 52.28   27.14   326.29  0.817   9.50 52.21   36.18   388.54  1.020
15 9.25 49.71   25.43   331.10  0.897   9.00 61.02   29.27   258.94
16 1.190   8.75 53.68   28.79   292.80  1.320   7.50 50.22   29.17
17 292.60  1.040   8.25 65.34   29.99   312.80  1.030   8.00 56.39
18 29.29   283.00  1.350   7.80 66.12   31.93   344.20  0.689   7.50
19 73.89   32.94   312.50  1.150   7.25 47.31   28.55   294.70  0.838
20 7.00 ; run;
21
22 proc print data=hemo; run;
```

语句说明：

此处语句规律同前面各章，主要是利用Data步建立的血红蛋白数据集，为了确认数据无误，用Print 过程显示。

2. 变量选择

此处变量较多(四个)，可以考虑按一定准则进行变量筛选，确定模型。关于自变量“最优回归”子集的确定方法，大体分为三类：

A. 全子集法。

就是通过计算所有可能的回归子集，按模型优劣标准进行排序，确定最优子集，模型优劣标准在本章有详细叙述。全子集法计算量大，但是信息全面。其程序如下

SAS源码 7.11: 变量选择——全子集法

```
1 proc reg data=hemo;
2   model y=x1 x2 x3 x4 /selection=cp;
3   title Hemoglobin and Other elements;
4 run;
```

(1). "model y=x1 x2 x3 x4"用于指定模型;

(2). "/selection=cp"选项

用于指定各子集所对应的模型的优劣比较标准，从而可以对所有可能子集排出一个序来，从结果中可以看出从上到下的次序中cp值逐渐增大；后面所跟各选项"aic bic sbc pc cp adjrsq sse mse rmse;"列出了一些关于选择变量是所用到的准则；

(3). 可以选择R2选择法(RSQUARE)

在第二行中"selection= cp"语句可以改为"selection= rsquare"后面的准则不变；从模型语句中的各自变量所有可能子集中选出规定数目的子集，使该子集所构成的模型的决定系数R2最大。要注意：当观测点少、且模型语句中变量数目过多时，程序不能运行，因为过多变量使误差项无自由度，设计矩阵不满秩，所以最多只能从所有可能的变量中选择观测点数减1个变量放入模型。本法和其它方法分别是按不同标准选出回归模型自变量的最优子集，这类选变量法不是从所有可能形成的变量中，而仅仅从模袖量中穷举。本法的局限性在于：其一，当样本含量小于等于自变量(含交互作用项)个数时，只能在一定数目的变量中穷举，为找到含各种变量数目的最优子集，要么增加观测，要么反复给出不同模型；其二，选最优子集的标准是R2，完全没考虑其他标准。

(4). 可以选择修正R2选择法(ADJRSQ)

在第二行中"selection= cp"语句可以改为"selection= adjrsq"后面的准则不变；根据修正的决定系数R2取最大的原则，从模型的所有变量子集中选出规定数目的子集。程序能运行的条件是设计矩阵X满秩。本法的局限性与上一方法相似：其一，与上面的方法方中"其一"相同；其二，选最优子集的标准只是用修正的R2取代未修正的R2而已，完全没考虑其他标准。

(5). Mallow's Cp选择法(CP)

根据Mallow's Cp统计量，从模袖量子集中选出最优子集。Cp统计量的数值比上面2种方法更大地依赖于MODEL语句所给出的模型，它比前二者多考虑的方面是：用模型语句决定的全回归模型估计出误差平和。程序能运行的条件是设计矩阵满秩。

B. 最优子集的简便方法。

包括向前选择法、向后删除法以及逐步回归法，这些是按一定标准指导下遵循某一个变量选择的路径，计算量较小，但是不像全子集法那般信息全面。

SAS源码 7.12: 变量选择——向后删除法

```
1 proc reg data=hemo;
2   model y=x1 x2 x3 x4 /selection=backward
```

```
3    sls=0.05;  
4    run;
```

(1). "selection=backward"表明使用的方法为"后退法"; "sls=0.05"指定了在剔除变量时所使用的显著性水平, 此处 $\alpha = 0.05$ 。

(2). 也可以用向前选择法, 此时"selection=forward", 也可以用"sle=0.05"指定了在选择变量时所使用的显著性水平, 此处 $\alpha = 0.05$, 如下。

SAS源码 7.13: 变量选择——向前选择法

```
1  proc reg data=hemo;  
2    model y=x1 x2 x3 x4 /selection=forward  
3    sle=0.05;  
4  run;
```

或者用逐步回归法

SAS源码 7.14: 变量选择——逐步回归法

```
1  proc reg data=hemo;  
2    model y=x1 x2 x3 x4 /selection=stepwise  
3    sls=0.05 sle=0.05;  
4  run;
```

此时, 需要"sle=0.05"指定了在选择变量进入时所使用的显著性水平, "sls=0.05"指定了剔除变量时所使用的显著性水平, 此处均为 $\alpha = 0.05$ 。

C. 计算量适中的选择法。

该法没有计算所有回归子集, 间于上述两者之间, 包括最小 R^2 增量法以及最大 R^2 增量法。

SAS源码 7.15: 变量选择——最大增量法

```
1  proc reg data=hemo;  
2    model y=x1 x2 x3 x4 /selection=MAXR;  
3  run;
```

其中也可以用最小增量法 "selection=minr"。

最后, 需奥指出的是, 无论这三类方法中哪一类, 如果需要指定某一个或几个特别的变量一定要包含, 例如, 此处用逐步回归法, 指定x4一定要包含, 则可以用如下代码

SAS源码 7.16: 变量选择——指定某变量

```
1  proc reg data=hemo;  
2    model y=x4 x1 x2 x3 /selection=stepwise  
3    include=1 sls=0.05 sle=0.05;  
4  run;
```

注意到"include=n"语句用于指定要包含的变量必须处于前n个位置, 此处"n=1"代表第一个变量位置, 注意前面的"model"顺序进行了调整, 要包含的变量一定要放在第一个位置上, 与"1"对应, 如果包含多了变量, 依此类推, 如果要包含的变量没有放在前边相应的位置, 则SAS系统按前边指定位置保留变量, 最终不能完成设计意图。

§7.3 菜单实现方法

§7.3.1 基本多元回归分析

下面使用“分析家”(Analyst)对话框实现多元回归分析过程。

若假设已经建立数据集 huishou.sas7bdat, 则类似第一章的相应部分打开 Analyst 对话框并打开数据集 huishou.sas7bdat, 再通过以下几个步骤进行多元回归分析。

(1). 打开多元回归分析对话框

Statistics \Rightarrow Regression \Rightarrow Linear ...

(2). 建立基本理论模型—选择因变量、自变量

在弹出的多元线性回归对话框中选择因变量($y \Rightarrow$ Dependent)与自变量($x1, x2 \Rightarrow$ Independent)。如图 7.1 所示。

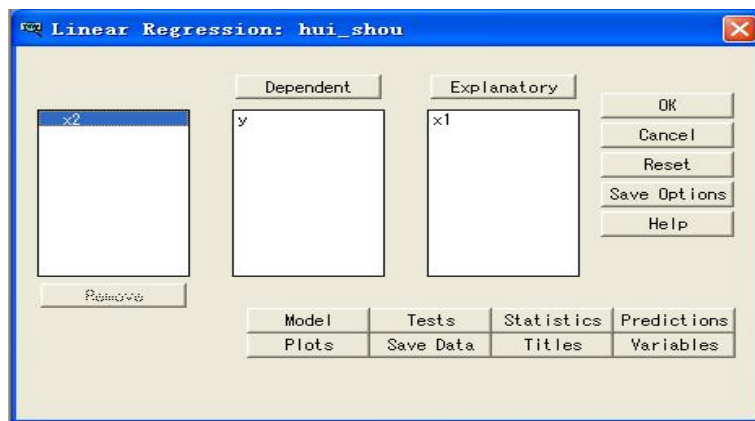


图 7.1: 简单回归分析对话框

(3). 其他回归分析选项

在多元线性回归对话框下方有几个按钮: **Model** 按钮让用户选择筛选自变量的方法, 默认为选择全模型; **Tests** 按钮让用户选择是否进行功效检验; **Statistics** 按钮让用户选择希望计算的统计量, 如参数估计, 标准回归系数, 估计的相关阵及协差阵等; **Predictions** 按钮让用户选择预测内容; **Plots** 按钮让用户常见绘图, 如: 因变量对自变量的散点图, 残差图, 影响分析等等, 譬如 \Rightarrow **Plot Options** \Rightarrow **Predicted** 选择绘制因变量对自变量的散点图, 并附上预测的置信限 \Rightarrow **OK Options**。

(4). 分析

将上述选项完成后, 从相应子对话框 \Rightarrow **OK** 返回简单线性回归主对话框, \Rightarrow **OK** 按钮, 系统对用户需求进行分析计算。

(5). 查看结果

从“分析家”的数据窗口左边的树状表可以选择你想查看的各类计算结果。

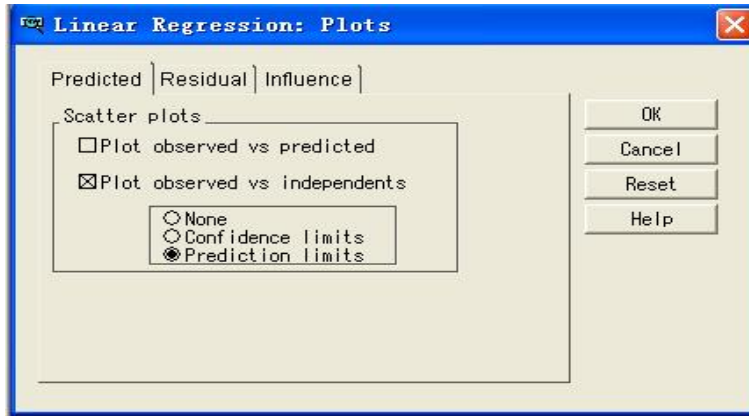


图 7.2: 多元线性回归的图形选项窗口

§7.3.2 基本诊断分析

下面使用“分析家”(Analyst)对话框实现多元回归的诊断过程。

若假设已经建立数据集, 先打开 Analyst 对话框并打开数据集, 再通过以下几个步骤进行诊断。

1. 残差诊断

(1). 打开多元回归分析对话框

Statistics \Rightarrow Regression \Rightarrow Linear ...

(2). 建立基本理论模型—选择因变量、自变量

在弹出的多元线性回归对话框中选择因变量(y \Rightarrow Dependent)与自变量($x_1, x_2 \Rightarrow$ Independent)。

(3). 其他回归分析选项

在多元线性回归对话框下方选择 Plots 按钮选择需要的某一类残差图, 譬如 \Rightarrow Plot Options \Rightarrow Residual 选择绘制残差对预测量的散点图。 \Rightarrow OK Options。

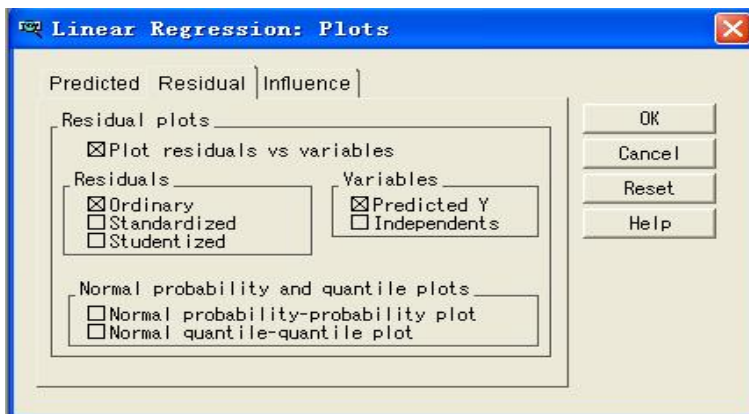


图 7.3: 多元线性回归的图形选项窗口

(4). 分析

将上述选项完成后，从相应子对话框 \Rightarrow 返回简单线性回归主对话框， \Rightarrow 按钮，系统对用户需求进行分析计算。

(5). 查看结果

从“分析家”的数据窗口左边的树状表可以选择你想查看的各类计算结果。

2. 共线性(异方差，自相关)诊断

打开多元回归分析对话框，并选择因变量、自变量建立基本理论模型后，从多元线性回归对话框下方作选择

(1). 回归分析选项

在多元线性回归对话框下方选择按钮 ，然后在弹出的统计量对话框中选择 \Rightarrow ，如图 7.4，让用户选择希望进行的检验，如共线性检验，也可以选择异方差检验和自相关检验等等； \Rightarrow 。

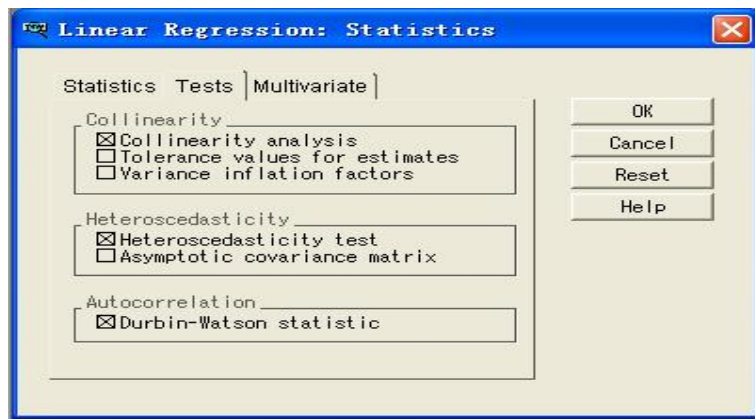


图 7.4: 统计量选项窗口

(2). 分析

将上述选项完成后，从相应子对话框 \Rightarrow 返回多元性回归主对话框，最后在主对话框确认操作 \Rightarrow 按钮，系统对用户需求进行分析计算。

(3). 查看结果

从“分析家”的数据窗口左边的树状表可以选择你想查看的各类计算结果。

§7.3.3 变量选择

下面使用“分析家”(Analyst)对话框实现多元回归分析过程。

若假设已经建立数据集 huishou.sas7bdat，则类似第一章的相应部分打开 Analyst 对话框并打开数据集 huishou.sas7bdat，再通过以下几个步骤进行多元回归分析。

(1). 打开多元回归分析对话框

\Rightarrow \Rightarrow

(2). 建立基本理论模型—选择因变量、自变量

在弹出的多元线性回归对话框中选择因变量(\Rightarrow)与自变量(\Rightarrow)。如图 7.5 所示。

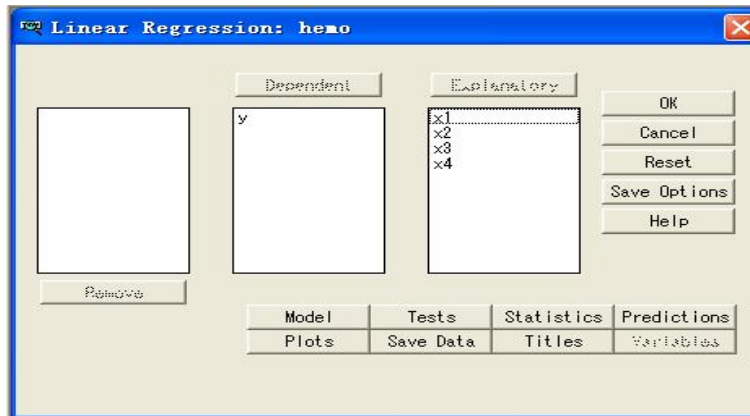


图 7.5: 多元回归分析对话框

(3). 指定最优变量子集选择方法

正如前边所说，在多元线性回归对话框下方有几个按钮，其中第一个按钮 按钮让用户选择筛选自变量的方法。如下图 7.6 所示，包含三个按钮：第一个为“Method”，指定变量选择的方法，主要就是前面所述的三类方法，默认为选择全模型(Full Model)；第二个按钮“Criterion”，指定输入输出的显著性水平sls,sle；第三个按钮是“Include”按钮，指定需要固定包含的变量；第四个按钮是“Statistics”，指需要输出哪些拟合优度统计量。

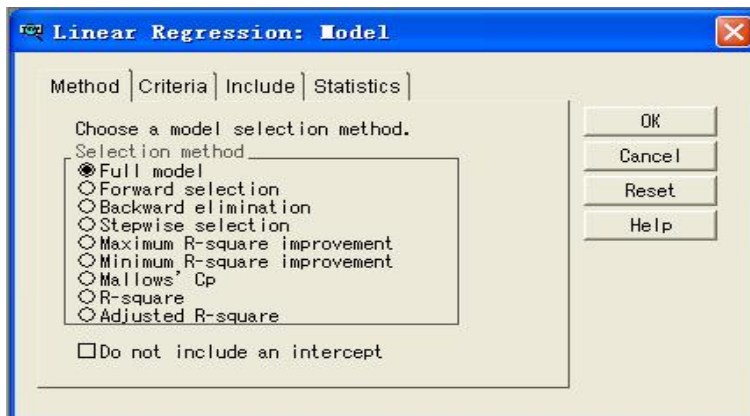


图 7.6: 多元回归分析“Model”子对话框

(4). 其他各种分析按钮

在多元线性回归对话框下方还有几个按钮，可以分别进行选择

(5). 分析

将上述选项完成后，从相应子对话框 \Rightarrow 返回线性回归主对话框， \Rightarrow 按钮，系统对用户需求进行分析计算。

(6). 查看结果

从“分析家”的数据窗口左边的树状表可以选择你想查看的各类计算结果。

第八章

方差分析模型(ANOVA)与试验设计问题

§8.1 方差分析引论

§8.1.1 理论模型的建立

一 背景以及常见实例分析

1. 试验目的

在科学的各个领域和生产的各行各业经常要做试验，试验的目的一般有三：

- (1). 可能是比较某个现象中的各个因素的重要性；
- (2). 以及它们的不同状态的效果；
- (3). 也可能是要寻找某个特定过程中各个变量之间的数量规律。

2. 常见实例说明

例 8.1.1 某个医师希望对治疗扁桃体的 2 种不同药物——洁霉素和青霉素的疗效进行比较。试验的目的是要确定哪种药物的疗效好。

解： 试验可按如下方法进行：假设我们采取双盲试验法，即病人不知道自己服用三种药中的哪一种，医生也不知道哪个病人服用哪种药，只有试验设计和分析者掌握情况，此举可以排除试验者以及受试者的心理因素对试验结果的影响(属于一种处理影响要素的一种方法——切断法——后面专门讨论之)；另外然后选取扁桃体炎患者若 n 人，分别用两种药物治疗，并用 y_{ij} 表示服用第 i 种药的第 j 个病人的药效测量值，则 y_{ij} 可表示为：

$$y_{ij} = \mu + \alpha_i + \varepsilon_{ij}, i = 1, 2; j = 1, 2, \dots, n. \quad (8.1)$$

然后比较两种药物的平均疗效，其中 μ 称为总平均， α_i 表示第 i 种药的效应， ε_{ij} 表示随机误差，其均值为 0，方差相等但未知为 σ^2 ，彼此线性无关。

在这个问题中，我们感兴趣的因素(或因子)只有一个，即药品，它有三个不同的品种，称这三个品种为因子的“水平”(Level)或“处理”(Treatment) 上述模型 (8.1) 称为“单向分类模型”(或“单因素分类模型”)，若用矩阵符号

$$\begin{pmatrix} y_{11} \\ \vdots \\ y_{1n} \\ y_{21} \\ \vdots \\ y_{2n} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 \\ \vdots & \vdots & \vdots \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ \vdots & \vdots & \vdots \\ 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} \mu \\ \alpha_1 \\ \alpha_2 \end{pmatrix} + \begin{pmatrix} \varepsilon_{11} \\ \vdots \\ \varepsilon_{1n} \\ \varepsilon_{21} \\ \vdots \\ \varepsilon_{2n} \end{pmatrix} \quad (8.2)$$

用 $\mathbf{y}, \mathbf{X}, \boldsymbol{\beta}, \boldsymbol{\varepsilon}$ 分别表示上式中的四个向量或矩阵, 则上述模型有以下简洁的矩阵形式

$$\begin{cases} \mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon} \\ \boldsymbol{\varepsilon} \sim N(\mathbf{0}, \sigma^2 \mathbf{I}_n) \end{cases} \quad (8.3)$$

上面的模型 (8.1)、(8.2) 或 (8.3) 均等价, 与前面所学的回归分析模型形式相同, 事实上它们同属于“线性模型”(Linear Model)的范畴, 不过也有其不同之处, 这种不同之处是本质的, 造成了设计矩阵的“列降秩”。事实上, 设计矩阵 \mathbf{X} 的元素只能取1或0两个值。

(1). 除了第一列以外, 设计阵 \mathbf{X} 的某列中的某个元素是1或0表示对应的病人服用了或者没服用该列对应的那种药。也就是说, 设计阵 \mathbf{X} 中的元素 $x_{ij}(j > 1)$ 只表示了实验某要素的某个处理效应的存在与否, 这是一种特殊的离散变量的“虚拟变量”化, 是以该因素各个处理均不出现为基础或标准的(注意: 与回归分析中引入虚拟变量的方式不同, 那儿是以某一个处理发生为基础相对而言的);

(2). 在模型 (8.3) 中, 设计阵的秩 $\text{rk}(\mathbf{X}) = 2 < 3$ 为列降秩的, 这是方差分析模型的一个特点, 因而从“线性模型”系数的可估性角度而言, 均是不可估的, 但是系数的某些线性组合却是可估的。

例 8.1.2 某个医师希望对治疗扁桃体炎的 a 种不同药物的疗效进行比较, 此处为了更为全面的了解病人情况, 使得具有代表性, 于是从 b 家不同医院(“多中心”)分别获得病人疗效数据。试验的目的是要确定哪种药物的疗效好。

解: 影响疗效 y 的有两个因素 A, B , 设 A 有 a 个水平, B 有 b 个水平, 并用 y_{ij} 表示在 A 的第 i 个水平以及 B 第 j 个水平下所形成的“组合水平”(容易计算总共有 $a \times b$ 个组合水平)的药效测量值(那就意味着每一个组合水平下都未有重复), 则 y_{ij} 可表示为:

$$y_{ij} = \mu + \alpha_i + \beta_j + \varepsilon_{ij}, i = 1, 2, \dots, a; j = 1, 2, \dots, b. \quad (8.4)$$

然后比较两种药物的平均疗效, 其中 μ 称为总平均, α_i 表示 A 的第 i 个水平的效应, β_j 表示 B 的第 j 个水平的效应, ε_{ij} 表示随机误差, 其均值为0, 方差相等但未知为 σ^2 , 彼此线性无关。

在这个问题中, 出现(但并非都是我们感兴趣的, 中心要素加入是被迫的, 药品才是感兴趣的)了两个因素(或因子)上述模型 (8.1) 称为“两向分类模型”(或“两因素分类模型”), 若用矩阵符号

$$\begin{cases} \mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon} \\ \boldsymbol{\varepsilon} \sim N(\mathbf{0}, \sigma^2 \mathbf{I}_{ab \times ab}) \end{cases} \quad (8.5)$$

其中 $\mathbf{y}, \mathbf{X}, \boldsymbol{\beta}, \boldsymbol{\varepsilon}$ 分别表示为

$$\begin{aligned} \mathbf{y} &= (y_{11}, \dots, y_{1b}, y_{21}, \dots, y_{2b}, \dots, y_{a1}, \dots, y_{ab})^T \\ \boldsymbol{\beta} &= (\mu, \alpha_1, \dots, \alpha_a, \beta_1, \dots, \beta_b)^T \\ \boldsymbol{\varepsilon} &= (\varepsilon_{11}, \dots, \varepsilon_{1b}, \varepsilon_{21}, \dots, \varepsilon_{2b}, \dots, \varepsilon_{a1}, \dots, \varepsilon_{ab})^T \end{aligned} \quad (8.6)$$

，此处需要特别注意的是设计阵的写法：

$$\begin{aligned} \mathbf{X} &= \begin{pmatrix} \mathbf{1}_b & \mathbf{1}_b & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{I}_b \\ \mathbf{1}_b & \mathbf{0} & \mathbf{1}_b & \cdots & \mathbf{0} & \mathbf{I}_b \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{1}_b & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{1}_b & \mathbf{I}_b \end{pmatrix} \\ &= (\mathbf{1}_{ab} : \mathbf{I}_a \otimes \mathbf{1}_b : \mathbf{1}_a \otimes \mathbf{I}_b) \end{aligned} \quad (8.7)$$

上面的模型 (8.4) 或 (8.5) 均等价，与前面所学的回归分析模型形式相同，事实上它们同属于“线性模型”(Linear Model)的范畴，不过也有其不同之处，这种不同之处是本质的，造成了设计矩阵的“列降秩”。事实上，设计矩阵 \mathbf{X} 的元素只能取1或0两个值。

(1). 除了第一列以外，设计阵 \mathbf{X} 的某列中的某个元素是1或0表示对应的病人服用了或者没服用该列对应的那种药。也就是说，设计阵 \mathbf{X} 中的元素 $x_{ij}(j > 1)$ 只表示了实验某要素的某个处理效应的存在与否，这是一种特殊的离散变量的“虚拟变量”化，是以该因素各个处理均不出现为基础或标准的(注意：与回归分析中引入虚拟变量的方式不同，那儿是以某一个处理发生为基础相对而言的)；

(2). 在模型 (8.5) 中，设计阵的秩 $\text{rk}(\mathbf{X}) = a + b - 1 < a + b + 1$ 为列降秩的，这是方差分析模型的一个特点，因而从“线性模型”系数的可估性角度而言，均是不可估的，但是系数的某些线性组合却是可估的。

此处除了考虑药品对疗效的影响，一般都是指定若干家医院进行临床试验，然而医院自身的医疗条件以及声望等等可能成为影响要素，因而进行分层抽样，也要考虑医院的要素，这种要素称为区组要素。试验可按如下方法进行：假设我们采取双盲试验法，即病人不知道自己服用三种药中的哪一种，医生也不知道哪个病人服用哪种药，只有试验设计和分析者掌握情况，此举可以排除试验者以及受试者的心理因素对试验结果的影响(属于一种处理影响要素的一种方法——切断法——后面专门讨论之)；控制其他各要素的影响；并采用随机方法(把其他为控制或者切断的要素纳入随机因素)，从各中心(假设共有 b 家医院或中心)随机选取扁桃体炎患者若 $a \times b$ 人(属于重复测量或者交叉试验，但是没有时间效应)，分别用 b 种药物治疗。

二 理论模型建立—影响因子的处理方法

1. 响应变量以及相关因子

上述模型中衡量试验结果好坏的指标称为“响应变量”(Response Variable)，很多情况下响应变量可用数值表示，称其为“定量的响应变量”，例如：例 (8.1.1) 中的疗效指标；然而也有某些响应变量不能用数值表示，该情况初等试验设计内容一般不涉及，需要用到广义线性模型，特别是著名的Logistic模型，。

我们将影响响应变量的因素称之为试验问题中的因子，因子的每个可能的取值称之为“水平”或者“处理”。统计模型中关于因子的处理方法很多，大体分为四类：

(1). 控制因子：某些因子如果无需考虑，或者说若只要考虑某一个特定水平下的关系时，该因子可以被“控制”起来；

(2). 切断因子：某些因子对响应变量的影响无需考虑，且能通过某种方法切断其影响机制，使其变为“无关因子”；

典型的切断方法就是生物制药试验中的“双盲试验”(Double-blind Trials), 这是为了切断试验者或者受试者的心理因素对试验产生的影响, 而被“盲”住而“切断”, 这种思想在其他场合也可以类似使用, 不过需要小心挖掘其作用机制。

(3). 自变量因子: 需要重点关注或者是试验者感兴趣的因子, 或者说随不感兴趣, 但是为了不造成无偏性或者异方差性, 需要引入的因子

(4). 随机因子: 某些因子无法控制或者无法考虑, 被迫“随机化”(Randomization)而进入随机误差部分(Stochastic Error Term), 注意到某些可观测因子, 若无需考虑, 也就进入随机部分。随机化是指试验仪器、材料、人员... 等的布置要随机的确定, 诸试验单元的执行顺序要随机的确定。由于使用了随机化这一手段, 就给分析试验结果时提供了使用统计方法的基础。主要体现在两个方面:

一方面, 此时试验结果是一个个随机变量, 随机化保证了其独立性, 在对效应作检验或估计时, 就可以用数理统计学中有关独立样本理论的方法的基础;

另一方面, 在分析试验结果时, 随机化可以有效消除“外来因子”对数据的干扰。例如: 在药物疗效试验时, 不使用随机化的话, 可能导致一种药只对中青年扁桃体炎换种使用, 另一种药只对老人、幼儿扁桃体炎患者使用, 使得个体差异这一“外来因子”和药物对疗效的影响混杂了, 给数据分析带来困难。使用随机化这一手段, 可以大体上保证两种药物都用到各种年龄、不同性别、各种健康状况的扁桃体炎患者身上, 使得两种药物在个体差异方面取得某种平衡, 在对两种药物的平均疗效作比较时, 消除个体差异这一“外来因子”对数据的干扰。

2. 统计模型的建立

During the discussion, questions were raised about cause and effect and chance occurrences, which may be summarized as follows: "You have emphasized the uncertainty of natural events. If events happen at random, how can we understand, explore and explain nature?"

I am glad this question is raised. Life would be unbearable if events occur at random in a completely unpredictable way and uninteresting, in the other extreme, if everything is deterministic and completely predictable. Each phenomenon is a curious mixture of both, which makes "life complicated but not uninteresting" (as J. Neyman used to say).

There are logical and practical difficulties in explaining observed phenomena and predicting future events through the principle of cause and effect.

Logical, since we can end up in a complex cause-effect chain. If A_2 causes A_1 , we may ask what causes A_2 . Say A_3 ; then what causes A_3 and so on. We may have an endless chain and at some stage the quest for a cause may become difficult or even logically impossible forcing us to model events at that stage through a chance mechanism.

Practical, since, except in very trivial cases, there are infinitely many (or finitely large number of) factors causing an event. For instance, if you want to know whether the toss of a coin results in a head or a tail you must know several things. First, the magnitudes of numerous factors such as initial velocity (x_1), measurements of the coin (x_2), nervous state of the individual tossing the coin (x_3), ..., which determine the event (y), head or tail, and then the relationship

$$y = f(x_1, x_2, x_3 \cdots) \quad (8.8)$$

must be known. Uncertainty arises if $f(\cdot)$ is not known exactly, if the values of all the factors x_1, x_2, \dots cannot be ascertained and if there are measurement errors. We may have information only on some of the factors, say x_1, \dots, x_n , forcing us to model the outcome y as

$$y = f_a(x_1, \dots, x_p) + \varepsilon \quad (8.9)$$

where $f_a(\cdot)$ is an approximation to f and ε is the unknown error arising out of our choice of $f_a(\cdot)$, lack of knowledge on the rest of the factors and measurement errors. Modeling for uncertainty in the choice of f_a and the error ε through a chance mechanism becomes a necessity.


三 方差分析的基本步骤

1. 问题的叙述
2. 试验计划的设计与实施
3. 试验结果的统计分析
4. 统计分析报告


§8.1.2 新药的临床试验

一 新药临床试验


定义 8.1 新药临床试验(Clinical Trials)是指对任何在人体(病人或者健康志愿者)中进行试验的药物的系统性研究,以证实或者揭示试验用药品的作用(疗效)、不良作用及(或)试验用药品的吸收、分布、代谢和排泄情况,目的是确定试验用药品的疗效和安全性。

 **新药分类:** 指尚未在中国境内上市的药物,包括中药、天然药物和化学药物以及生物制品。

- (1). 中药与天然药物分类
- (2). 化药分类
- (3). 生物制品分类

 **试验目的:**


- (1). 疗效
- (2). 安全性
- (3). 代谢机制: 吸收、分布、代谢与排泄规律

 **试验对象:** 严格的对象是“病人”或“患者”。

所以以下几类不属于或者不严格属于“新药临床试验”:

(1). 动物实验: 不属于新药临床试验,但是却能够作为人体试验的前期,为之提供致死剂量等多方面信息。

(2). 健康志愿者: 严格来说也不属于新药临床试验,但是更接近于病患,因而为之提供致死量、剂量等重要关键指标信息,可以视为新药临床试验边缘。

 **考虑要素:** 只对药品类型、剂量等药品相关指标感兴趣,由此为了提高精度和减少偏差可以引入其他因素,如: 减少偏差引入中心要素等。

✎ 新药临床试验的分期问题：新药临床试验分为I、II、III、IV期，各期试验的目的、对象等都有差异，从新药试验早期到后期构成完整的新药试验。

(1). I期临床试验：

初步的临床药理学及人体安全性评价试验，观察人体对于新药的耐受程度和药物代谢动力学，为制定给药方案提供依据。

(2). II期临床试验：

随机盲法对照临床试验。对新药有效性及安全性做出初步评价，推荐临床给药剂量。

(3). III期临床试验：

扩大的多中心临床试验。应遵循随机对照原则，进一步评价有效性、安全性。

(4). IV期临床试验：

新药上市后监测。在广泛条件下考察疗效和不良反应(注意罕见不良反应)。

✎ 临床试验的对象与目的：

临床试验系指任何一种有病人(受试者)参加的有计划的试验，目的是寻求在相同条件下，对未来病人的一种最合适的治疗方法。具有以下特点：

(1). 试验对象：一般受试者。

(2). 试验目的：治疗方案

(3). 试验要素：药品剂量、类型等，关心其有效性指标或者其他指标。

新药临床试验属于一种特殊的临床试验。

✎ 一般试验设计问题：

(1). 动物试验

(2). 人体试验

二 多中心试验

1. 多中心试验

定义 8.2 多中心临床试验(Multi-center Clinical Trials)系指由一个或几个单位的主要研究者总负责，多个单位的研究者合作，按同一个试验方案同时进行的临床试验。主要研究者所在的单位在中国俗称为“组长单位”或“牵头单位”。

✎ 多中心试验的必要性：

(1). 数据的速度和样本量：快、多、大、短

收集病例快，病例多，试验规模大，完成临床试验时间短。

(2). 数据的代表性：适应面广、可信度大

由于涉及多个中心的医务人员和病人，所获结论就有比较广泛的意义，减少偏度，适应面广，可信度大。

(3). 集思广益：

✎ 多中心试验的缺点：

✎ 多中心试验的数据质量控制：

(1). 操作误差控制—人员培训

(2). 数据的截尾与删失—病人进入与病人退出

(3). 异常值检测—一致性检验

2. 统计模型


定义 8.3 所谓模型(Model)是指在某些理论以及假设的基础上, 对实际对象的一些主要因素(或指标, 或基本量)的相互关系作出的描述。在多中心试验资料的统计分析中使用的是统计模型, 在研究计划中需要写明使用的是何种统计模型。多中心临床试验的统计模型中需要考虑中心的效应。

例 8.1.3

三 双盲试验

1. 双盲的定义

定义 8.4 “双盲试验”系指试验中受试者、研究者、参与疗效和安全性评价的医务人员、监察员、数据管理人员及统计分析人员都不知道治疗分配程序, 即: 哪个病人被分入哪一组。


 而广义的双盲试验根据“盲”的深度, 有其他称呼: 从

(1). 双盲试验: 狭义角度而言, “双盲试验”(Double Blind Clinical Trials)指的是受试者和试验者都不知道治疗分配程序;


(2). 三盲试验: 增加参与疗效和安全性评价的医务人员的“盲”

(3). 四盲试验: 增加监察员的“盲”

(4). 五盲试验: 增加数据管理人员以及统计分析员的“盲”

 双盲试验的目的: 避免偏倚(bias)

如果不进行双盲, 则因为受试者或者试验者的心理暗示称为疗效因素之后, 而不能考虑进入模型而产生的“模型偏差”。

 双盲试验的适应场合:

(1). 当一个临床试验, 反映疗效和安全性的主要变量(Primary Variable or Primary Endpoints)是一个受主观因素影响较大的变量时, 例如: 神经精神病科的各种量表(MMSE量表, 神经功能缺损量表, 生活力量表等), 又如在某些情况下需用临床全局评价指标(Clinical Global Assessment)评价疗效和安全性时, 这时必须使用双盲试验。

(2). 即使一个主要变量为客观指标(如: 生化指标、血压测量值等), 为科学、客观的评价疗效和安全性也应该使用双盲设计, 因为可能出现申报者任意选择和挑选病例的情况以及修改病例报告表, 双盲能有效避免这种情况出现。

2. 双盲试验的实施步骤

STEP 1: 抽样框的确定

(1). 如考虑多中心试验, 可以确定一个分层抽样框(Sampling Frame)

(2). 还可以进一步考虑“区组效应”。


STEP 2: 随机编码的产生

STEP 3: 分配


四 对照组的设定

1. 对照组

定义 8.5 对照组是处于与试验组同样条件下的一组受试者，对照组和试验组唯一的差别是试验组接受新药治疗，对照组则是接受对照药物的治疗。


 设置对照组的意义：

比较研究是临床试验的重要方法，为了说明一个新药的疗效和安全性，必须有供比较的对照组。

 科学设置对照组的统计要求——可比性要求：

临床试验要求试验组与对照组来自相同的受试者总体。不但在试验开始时，两组受试者基本情况(Baseline)是相应的或者相似的，而且要求在试验进行中除了试验药物不相同外，其他条件均需保持一致。

如果两组病人条件不一致，就会造成试验中的“偏倚”(Bias)，影响到分析结果的解释(异方差导致)，同时所估计的处理效应(Treatment Effect)会偏离真正的效应值。

 设立对照组的必要性与强制性：

(1). 必要性：因果推断的合理述求，是逻辑推理的必然要求。

(2). 强制性：国家药品监督管理局1999年5月1日颁布的新药审批办法，第四章新药的临床研究中第12条规定“新药的二期临床试验为随机盲法对照试验，三期临床试验为扩大的多中心临床试验，应遵循随机对照的原则”。

2. 对照组的类型

临床试验中的对照组设置有以下五种类型，前四种需要试验组和对照组均来自于同一个病人总体，而且随机的进入到各个组别；第五种外部对照则是对照受试者来自于与试验组不同的病人总体，它只是适应于一些特殊目的或特殊情况的试验。

(1). 安慰剂对照(Placebo Control)：

是一种伪装物(Dummy medication)，其外观如剂型、大小、颜色、重量、气味以及口味等都尽可能与试验药物一致，但不含有药物的有效成分；

安慰剂对常常是双盲试验，当然可以是交叉试验或者平行试验。

(2). 空白对照(No-treatment Control)：

临床试验中选定的对照组并未加以任何对照药物称为空白对照。空白对照组与试验组的受试者分配必须遵循随机化原则，以保证其来自于统一统计总体。

空白对照不给病人任何药物，因而是非盲的(有区别，无法盲)，当然可以交叉试验或者平行试验。

(3). 剂量反应对照(Dose-response Control)：

(4). 阳性药物对照(Active Control)：

(5). 外部对照(External Control)：

对照可以是平行对照，也可以是交叉对照；可以是盲法，也可以是非盲法；同一个临床试验可以采用一个或多个类型的对照组形式，须视具体情况，试验目标而定。所谓平行对照是指将来自于同一个总体的受试者，随机的分成 K 个组别，各组别不但在试验前同质，而且在试验中处于相同的条件，唯一的不同就是各个组别所施加的处理不同，

如有的组别为试验药组, 有的为安慰剂组, 根据试验结果, 比较各个组别。而交叉对照系指同一个受试者在不同的时期, 给以不同的处理, 而每个时期都经历清洗期以保持给予不同处理前, 受试者保持相同的状态, 再根据一定数量的受试者、试验结果比较各组别。

§8.2 单因子试验

§8.2.1 单因子试验的统计模型

设因子 A 有 a 个不同的水平(或处理), 分别记为 A_1, A_2, \dots, A_a , 试验的目的在于比较这 a 个水平的优劣。如果试验中不存在误差或者误差很小, 则由各个水平下的一个试验观测值就可对诸水平进行比较, 但实际上, 每个试验的观察值都都不可避免的存在着随机误差, 因此应通过试验的重复获得误差的估计, 然后再对诸水平作比较。设在水平 A_i 下作了 n_i 次重复试验 $i = 1, 2, \dots, a$, 则总共要做 $n = \sum_{i=1}^a n_i$ 次试验, 为了使得原料、设备……等方面对所采用的诸水平而言尽可能保持平衡, 这 n 个试验应按随机的顺序进行, 这种试验设计的技术称为“完全随机化设计”。

设 y_{ij} 为第 i 个水平 A_i 所重复进行的第 j 次试验, 其统计模型为

$$\begin{cases} y_{ij} = \mu + \alpha_i + \varepsilon_{ij}, \\ \varepsilon_{ij} \text{ i.i.d. } \sim N(0, \sigma^2), i = 1, \dots, a; j = 1, \dots, n_i, \\ \text{s.t. } \sum_{i=1}^a n_i \alpha_i = 0. \end{cases} \quad (8.10)$$

即: y_{ij} 是一般平均(总平均, 综合平均)、第 i 个水平 A_i 的效应以及随机误差三者之和, 其中约束不成立, 即 $\sum_{i=1}^a n_i \alpha_i = d \neq 0$, 此时只需要作变换 $\mu^* = \mu + d/n, \alpha_i^* = \alpha_i - d/n$

分别取代上式中的 μ, α_i 则有新模型 $y_{ij} = \mu^* + \alpha_i^* + \varepsilon_{ij}$ 满足 $\sum_{i=1}^a n_i \alpha_i^* = 0$ 。

若令

$$\begin{aligned} \mathbf{y} &= (y_{11}, \dots, y_{1n_1}, y_{21}, \dots, y_{2n_2}, \dots, y_{a1}, \dots, y_{a,n_a})^T \\ \boldsymbol{\beta} &= (\mu, \alpha_1, \dots, \alpha_a, \beta_1, \dots, \beta_b)^T \\ \boldsymbol{\varepsilon} &= (\varepsilon_{11}, \dots, \varepsilon_{1n_1}, \varepsilon_{21}, \dots, \varepsilon_{2n_2}, \dots, \varepsilon_{a1}, \dots, \varepsilon_{a,n_a})^T \end{aligned} \quad (8.11)$$

以及设计矩阵

$$\begin{aligned} \mathbf{X} &= \begin{pmatrix} \mathbf{1}_{n_1} & \mathbf{1}_{n_1} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{1}_{n_2} & \mathbf{0} & \mathbf{1}_{n_2} & \cdots & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{1}_{n_a} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{1}_{n_a} \end{pmatrix} \\ &= (\mathbf{1}_n : \mathbf{I}_a \otimes \mathbf{1}_b) \end{aligned} \quad (8.12)$$

则单因素方差模型 (8.10) 可以写成:

$$\begin{cases} \mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon} \\ \boldsymbol{\varepsilon} \sim N(\mathbf{0}, \sigma^2 \mathbf{I}_n) \\ \text{s.t. } \sum_{i=1}^a n_i \alpha_i = 0. \end{cases} \quad (8.13)$$

☞ 方差分析模型的特点：

- (1). 自变量为虚拟变量，且以全未观察为基础或标准；
- (2). 上述第二部分决定了设计阵是“列降秩”的，因而系数不可估。

☞ 方差分析模型的约束部分：

(1). 系数不可估，于是进行“平移”以方便估计，这种平移可以从正规方程组中看出来，主要是为了估计方便。

- (2). 有无穷多组解，平移之后只是获得了较快捷的一组，改组解是不可估的。

☞ 方差分析模型的分类：

- (1). 固定效应模型

如果因子是 A 的 a 个水平是由试验者主观愿望指定好的，并在试验中得到很好的控制，这种模型称之为“固定效应模型” (Fixed Effect Model)。估计以及检验结论只适用于试验所使用过的因子 A 的各个水平，而不能推广到试验中未使用过的水平。

- (2). 随机效应模型

如果因子 A 的 a 个水平是从因子 A 的全部可能的水平这一总体中选取的一个随机样本，这种情况下的统计模型称之为“随机效应模型” (Random Effect Model)

§8.2.2 固定效应模型的统计分析

一 参数估计

根据最小二乘原理，获得其正规方程为：

$$\begin{aligned} \mathbf{X}'\mathbf{X}\boldsymbol{\beta} &= \mathbf{X}'\mathbf{Y} \\ \begin{pmatrix} \mathbf{1}'_n \\ \mathbf{I}'_a \otimes \mathbf{1}'_b \end{pmatrix} (\mathbf{1}_n, \mathbf{I}_a \otimes \mathbf{1}_b) \boldsymbol{\beta} &= \mathbf{X}'\mathbf{Y} \\ \begin{pmatrix} \mathbf{1}'_n \mathbf{1}_n & \mathbf{1}'_n (\mathbf{I}_a \otimes \mathbf{1}_b) \\ (\mathbf{I}'_a \otimes \mathbf{1}'_b) \mathbf{1}_n & (\mathbf{I}'_a \mathbf{I}_a) \otimes (\mathbf{1}'_b \mathbf{1}_b) \end{pmatrix} \boldsymbol{\beta} &= \begin{pmatrix} y_{..} \\ y_{1.} \\ \cdots \\ y_{a.} \end{pmatrix} \end{aligned}$$

注意到矩阵 $\mathbf{X}'\mathbf{X}$ 是退化的，不能直接求逆。于是展开观察：

$$\begin{cases} n\mu + \sum_{i=1}^a n_i \alpha_i = y_{..}, \\ n_i \mu + n_i \alpha_i = y_{i.}, i = 1, 2, \cdots, a. \end{cases}$$

其中 $y_{..} = \sum_{i=1}^a \sum_{j=1}^b y_{ij}$, $y_{i.} = \sum_{j=1}^b y_{ij}$ 。

由于设计阵列降秩，因而 $\mathbf{X}'\mathbf{X}$ 退化，LSE不唯一，此时对任意 $c \in \mathcal{M}(\mathbf{X}')$, $c'\boldsymbol{\beta}$ 是可估函数，且 $c'\hat{\boldsymbol{\beta}}$ 是 $c'\boldsymbol{\beta}$ 的LSE，其中 $\hat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{X})^- \mathbf{X}'\mathbf{Y}$ 是任意LS解。即对可估函数而言，它的LSE 不依赖于LS解的选择。因此，我们只需求正则方程的任一特解即可。于是我们设立了上述边界条件(注意到这个边界条件所规定的解一定是某一个LSE解，由正规

方程观察可得):

$$\begin{cases} \hat{\mu} = \frac{1}{n} y_{..} = \bar{y}_{..}, \\ \hat{\alpha}_i = \frac{1}{n_i} y_{i.} - \hat{\mu} = \bar{y}_{i.} - \bar{y}_{..}, i = 1, 2, \dots, a. \end{cases}$$

注意到上述估计并非 $\mu, \alpha_i, i = 1, \dots, a$ 的无偏估计, 因为由估计理论可知这些系数均是不可估的。

因为 $\text{rk}(\mathbf{X}) = \text{rk}(\mathbf{X}') = a$, 所以此时仅有 a 个线性无关的可估函数 $\mathbf{c}'\hat{\boldsymbol{\beta}}$, 即: 存在 $\mathbf{c}_i \in \mathcal{M}(\mathbf{X}'), i = 1, 2, \dots, a$ 是 R^n 的子空间 $\mathcal{M}(\mathbf{X}')$ 中的一组基, 满足 $\mathbf{c}_i'\hat{\boldsymbol{\beta}}$ 可估。很容易发现这么一组可估函数 $\mu + \alpha_i = \mathbf{c}_i'\boldsymbol{\beta}, i = 1, \dots, a$ 其中 $\mathbf{c}_i = (1, 0, \dots, 0, 1, 0, \dots, 0)^T$ 为其相应的一组基。显然任意 $\mathbf{c} \in \mathcal{M}(\mathbf{X}')$ 均可表示成这组基的线性组合 $\mathbf{c} = \sum_{i=1}^a d_i \mathbf{c}_i$, 当然对其相应的可估函数:

$$\begin{aligned} \mathbf{c}'\hat{\boldsymbol{\beta}} &= \sum_{i=1}^a d_i \mathbf{c}_i' \hat{\boldsymbol{\beta}} \\ &= \sum_{i=1}^a d_i (\mu + \alpha_i) \\ &= \mu \sum_{i=1}^a d_i + \sum_{i=1}^a d_i \alpha_i \end{aligned}$$

如果想要得到一个只包含效应 α_i 而不包含总平均 μ 的可估函数, 当且仅当取 $\sum_{i=1}^a d_i = 0$, 于是有下面的定理:

定理 8.2.1 对于单向分类模型 (8.13), 有

- (1). $\sum_{i=1}^a d_i \alpha_i$ 可估 $\iff \sum_{i=1}^a d_i \alpha_i$ 是一个对照, 即: $\sum_{i=1}^a d_i = 0$;
- (2). 对照 $\sum_{i=1}^a d_i \alpha_i$ 的BLUE为 $\sum_{i=1}^a d_i \bar{y}_{i.}$.

二 方差分析

1. 假设检验问题

2. F-检验

三 多重比较方法

例 8.2.1 设有三种治疗方案, 18个病人随机地分配至各个治疗方案组中, 观测值见下表, 由于各个治疗方案是固定的, 故为固定作用模型。试分析各个治疗方案的疗效是否存在显著差异?

表 8.1: 各组的反应值

处理	反应值						
1	40	35	42	37	30	41	
2	-5	0	10	7	1	-3	-1
3	20	17	10	12	2		

解： 选定单因子固定效应模型

$$\begin{cases} y_{ij} = \mu + \alpha_i + \varepsilon_{ij}, \\ \varepsilon_{ij} \text{ i.i.d. } \sim N(0, \sigma^2), i = 1, 2, 3; j = 1, \dots, n_i, \\ \text{s.t. } \sum_{i=1}^3 n_i \alpha_i = 0. \end{cases} \quad (8.14)$$

其中 μ 为总平均, α_i 为第 i 套治疗方案的处理效应。

事实上所提问题相当于做了以下假设检验:

$$\begin{aligned} H_0 : \alpha_1 &= \alpha_2 = \alpha_3, \\ H_1 : \alpha_i, i &= 1, 2, 3 \text{ 不全相等.} \end{aligned} \quad (8.15)$$

运用SAS程序解决上述问题为。首先读入数据, 建立SAS数据集

```
data treatments; input treat response@@; cards;
  1 40 1 35 1 42 1 37 1 30 1 41
  2 -5 2 0 2 10 2 7 2 1 2 -3 2 -1
  3 20 3 17 3 10 3 12 3 2
; run; proc print; run;
```

■

然后利用GLM过程或者ANOVA过程进行处理

```
proc glm;
  class treat;
  model response=treat;
run;
proc anova;
  class treat;
  model response=treat;
run;
```

§8.2.3 随机效应模型的统计分析

- 一 试验设计与统计模型
- 二 统计分析

§8.2.4 模型诊断

- 一 方差齐性诊断
- 二 独立性诊断
- 三 正态性诊断

§8.3 多因子试验

§8.3.1 两因子试验的统计模型

§8.3.2 固定效应模型的统计分析

- 一 可加效应模型的统计分析
- 二 交互效应模型的统计分析

§8.3.3 随机效应模型与混合模型的统计分析

- 一 随机效应模型的统计分析
- 二 混合模型的统计分析

§8.3.4 多因子试验的设计与分析

- 一 统计模型
- 二 固定效应模型的统计分析
- 三 随机效应模型与混合模型的方差分析

§8.3.5 拉丁方设计与正交拉丁方设计

- 一 拉丁方设计及其统计模型
- 二 统计分析
- 三 希腊-拉丁方设计

§8.4 试验设计中的面板模型

§8.4.1 面板模型概述

§8.4.2 临床试验中的交叉试验

- 一 交叉试验及其步骤
- 二 2×2 交叉试验定量数据的统计分析
- 三 多个处理交叉试验定量数据的统计分析
- 四 2×2 交叉试验定性数据的统计分析

§8.4.3 重复测量设计

- 一 重复测量设计及其原理
- 二 单因子两组别模型
- 三 二因子单组别模型
- 四 协变量的校正

第九章

经典时序分析

§9.1 统计方法与实例分析

§9.2 程序实现方法

§9.3 菜单实现方法

第十章

Logistic回归与广义线性模型

§10.1 Logistic模型及其统计推断

§10.2 流行病学中的统计模型基础

§10.3 Logistic回归程序实现方法

§10.4 Logistic回归菜单实现方法

§10.5 广义线性模型

第十一章

多元数据处理

经典的多元统计分析理论，大体分为两类：分类方法和数据压缩方法。前者主要包括聚类分析和判别分析：聚类分析是“无监督学习”分类法，而“判别分析”则是“有监督学习”分类方法。数据压缩方法，通常指的就是“降维”（另一个意义指的就是样本压缩，如统计量的功能），主要也可以分为“无监督学习”降维和“有监督学习”降维，朱利平的“充分降维”就是有监督学习降维的典型代表；因子分析等是“无监督学习”降维的典型。

也特别需要注意的是，经典多元统计分析方法放在“数据挖掘”或者“统计学习”的背景下，已经与传统的统计学紧紧相连，并在更宽广的背景下激发了更丰富的思想和方法。例如：判别分析与回归模型特别是广义线性模型（如：Logistic模型）、神经网络、支撑向量机、决策树等等模型很难区分；例如：降维方法与有监督学习很难区分，并且称为特征提取与压缩的思想来源。所以，最佳思路就是将多元统计方法放在数据挖掘这个高度重新审视，交叉融合，以期获得新的生命力，这种期望也是非常合理的。

下面各节分别讲解各经典的多元统计方法，特别侧重于其基本步骤和算法，然后就是如何通过软件实现的方法和步骤。

§11.1 聚类分析及其实现

§11.1.1 统计方法与实例分析

§11.1.2 程序实现方法

§11.1.3 菜单实现方法

§11.2 判别分析及其实现

§11.2.1 统计方法与实例分析

§11.2.2 程序实现方法

§11.2.3 菜单实现方法

§11.3 主成分分析及其实现

§11.3.1 统计方法与实例分析

§11.3.2 程序实现方法

§11.3.3 菜单实现方法

§11.4 因子分析及其实现

§11.4.1 统计方法与实例分析

§11.4.2 程序实现方法

§11.4.3 菜单实现方法

§11.5 相关分析及其实现

§11.5.1 统计方法与实例分析

§11.5.2 程序实现方法

§11.5.3 菜单实现方法

第三部分 III

SPSS系统与数据分析

第十二章

SPSS语言介绍与系统简介

第十三章

多元统计分析

§13.1 回归分析及其实现

§13.1.1 统计方法与实例分析

§13.1.2 程序实现方法

§13.1.3 菜单实现方法

§13.2 主成分分析及其实现

§13.2.1 主成分分析

- 一 背景与基本思想
- 二 主成分的基本模型
- 三 主成分的计算方法
- 四 主成分的基本性质
- 五 主成分分析步骤

§13.2.2 实例分析与菜单实现方法

- 一 实例分析
- 二 菜单实现方法

§13.3 因子分析及其实现

§13.3.1 因子分析

- 一 背景与基本思想
- 二 因子分析的基本模型
- 三 因子分析的计算方法
- 四 因子分析的基本性质
- 五 因子分析步骤

§13.3.2 实例分析与菜单实现方法

- 一 实例分析
- 二 菜单实现方法

§13.4 聚类分析及其实现

§13.4.1 统计方法与实例分析

§13.4.2 程序实现方法

§13.4.3 菜单实现方法

§13.5 判别分析及其实现

§13.5.1 统计方法与实例分析

第四部分 IV

Minitab系统与数据分析

第十四章

Minitab系统介绍

第十五章

Minitab功能介绍

参考文献

- [1] Hansen, B. E. (2009). Econometrics[M]. unpublished manuscript.
- [2] 陈希孺, 王松桂(1987). 近代回归分析[M]. 合肥: 安徽教育出版社, 1987.
- [3] 陈希孺(1987). 数理统计引论[M]. 北京: 科学出版社.
- [4] 王松桂, 史建红等(2004). 线性模型引论[M]. 北京: 科学出版社.
- [5] 周纪芄(1993). 回归分析[M]. 上海: 华东师范大学出版社.