Lecture 14: SAS 简介

张伟平

Monday 14th December, 2009

Contents

Chapter 1
Introduction

SAS(Statistical Analysis System) 系统是用于数据分析和决策支持的大型集成式模块化软件包.

✿ 历史：最早由北卡罗来纳大学的两位生物统计学研究生编制, 并于1976年成立了SAS软件研究所, 正式推出了SAS软件。

✿ 在数据处理和统计分析领域, SAS系统被誉为国际上的标准软件系统, 并在96～97年度被评选为建立数据库的首选产品.

✿ 用户：全世界120多个国家和地区的近三万家机构所采用, 直接用户则超过三百万人, 遍及金融、医药卫生、生产、运输、通讯、政府和教育科研等领域.

✿ 案例：美国FDA新药审批程序==> SAS的权威地位

SAS Modules

最新版的SAS提供多达30多个模块(module). 比如

✺ **SAS/BASE** SAS 的核心，负责数据管理、交互应用环境管理、用户语言处理以 及调用其他SAS模块. SAS/BASE为SAS的数据库提供了丰富的数据管理功能, 还支 持标准的SQL语言对数据进行操作. SAS/BASE不仅能够制作简单的列表, 面且可以 制作比较复杂的统计报表. SAS/BASE还可以进行基本的描述性统计、相关系数的计 算，以及进行正态分布检验等.

✺ **SAS/STAT** SAS/STAT 覆盖了所有的实用数理统计分折方法, 是国际统计分析领域的标准软件. SAS/STAT 提供了十多个过程, 可进行各种不同模型或不同特点数据的回归分 析, 且具有多种模型选择方法. 可处理的数据有实型数据、有序数据和属性数据. 在方差分析方面,SAS/STAT 为多种试验设计模型提供了方差分 析工具. 另外, 它还有处理一般线性模型和广义线性模型的专用过程. 在多变量统 计方面, SAS/STAT 为主成分分析、典型相关分析、判别分析和因子分析提供了许多 专用过程. SAS/STAT 还包含多种聚类

准则的聚类分析方法.

✺ **SAS/IML** SAS/IML 提供功能强大的面向矩阵运算的编程语言, 帮助用户研究新算法或解决SAS中没有现成算法的专门问题. SAS/IML中的基本数据元素是矩阵, 它包含 大量的数学运算符, 函数和例行程序, 用户使用很少的语句便可以执行复杂的计算过程.

✺ **SAS/GRAPH** 图形模块. SAS/GRAPH 可将数据及其包含着的深层信息以 多种图形生动地呈现出来, 如直方图、饼分图、星形图、散点图、曲线图、三维曲面团、等高线图及地理图等. SAS/GRAPH 提供一个全屏幕编辑器, 提供多种设备程序, 支持非常广泛的图形输出设备.

✺ **SAS/ACCESS** 外部数据库接口模块, 提供了与大多数数据库管理系统连接 的接口, 并且其自身也能进行数据管理.

其他的模块如SAS/计量经济学和时间序列分析, SAS/运筹学, SAS/投资分析, SAS/实验设计, SAS/市场研究, SAS/项目管理等等.

SAS 特点

1. 信息存储简便灵活

SAS能和绝大多数数据库系统交换数据信息,具有很强的数据共享能力. 数据导入后, 分析功能非常强大.

2. 语言编程能力强

SAS 语言功能强大, 内含100多种函数和丰富的算术逻辑运算符 值语句、条件语句、数组和循环语句等对变量进行各种操作.

SAS语言有两类, 即**DATA语句**和**PROC语句**. SAS程序以DATA语句开始, 输 入或处理有关的数据集(SAS的数据文件). 让系统读入有关数据. DATA语句的部分 叫数据步. PROC 语句用于指出对哪种数据进行分析. PROC 语句的部分叫过程步, 利用系统提供的过程(SAS 提供的数据分析子程序)进行分析.

3. 丰富的统计分析方法

统计分折是SAS的主要功能. 它几乎囊括了所有的统计方法, 从基本统计到

多 元分析, 从预测预报到运筹决策, 从时序分析到经济计量等.

SAS 只需通过菜单的选择、对话框的操作告诉系统要做 什么, 而无须告之怎样做. 只要初步了解统计分析原理, 无须通统统计分析的各种算 法, 即可得到统计分析结果.

4. 较强的统计报表与绘图功能

SAS 支持绘制1－3维表格，它既可以各种格式输入数据, 也可以任何方式输出表格. SAS 可以绘制各种统计图形.

5. 友好的用户界面

SAS 提供了一系列灵活的菜单驱动, 面向任务的图形界面, 初学者和熟练的SAS 用户都可以全面地进行以数据为中心的基本处理. SAS 采用多窗口的操作环境和显 示方式, 远行一个SAS程序时, 用户可以从不同的窗口中了解程序的运行情况、出错 信息和输出结果. 对于数据和文件, 也各有窗口管理和显示.

6. 宏功能

SAS 有较强的宏代换功能. 如果用户需要多次做类似的工作, 其中仅是参数不 同, 则可以使用宏功能定义宏体, 在宏体中可以使用宏变量. 随后, 用户就可以使用 不同的参数载人宏体, 从而大大简化了程序的编写.

7. 支持分布式处理

SAS 支持分布式处理, 多台计算机协同工作, 有效减少了大数据量的统计分析所 需的时间, 并能充分享用网络环境中的软件及硬件资源.

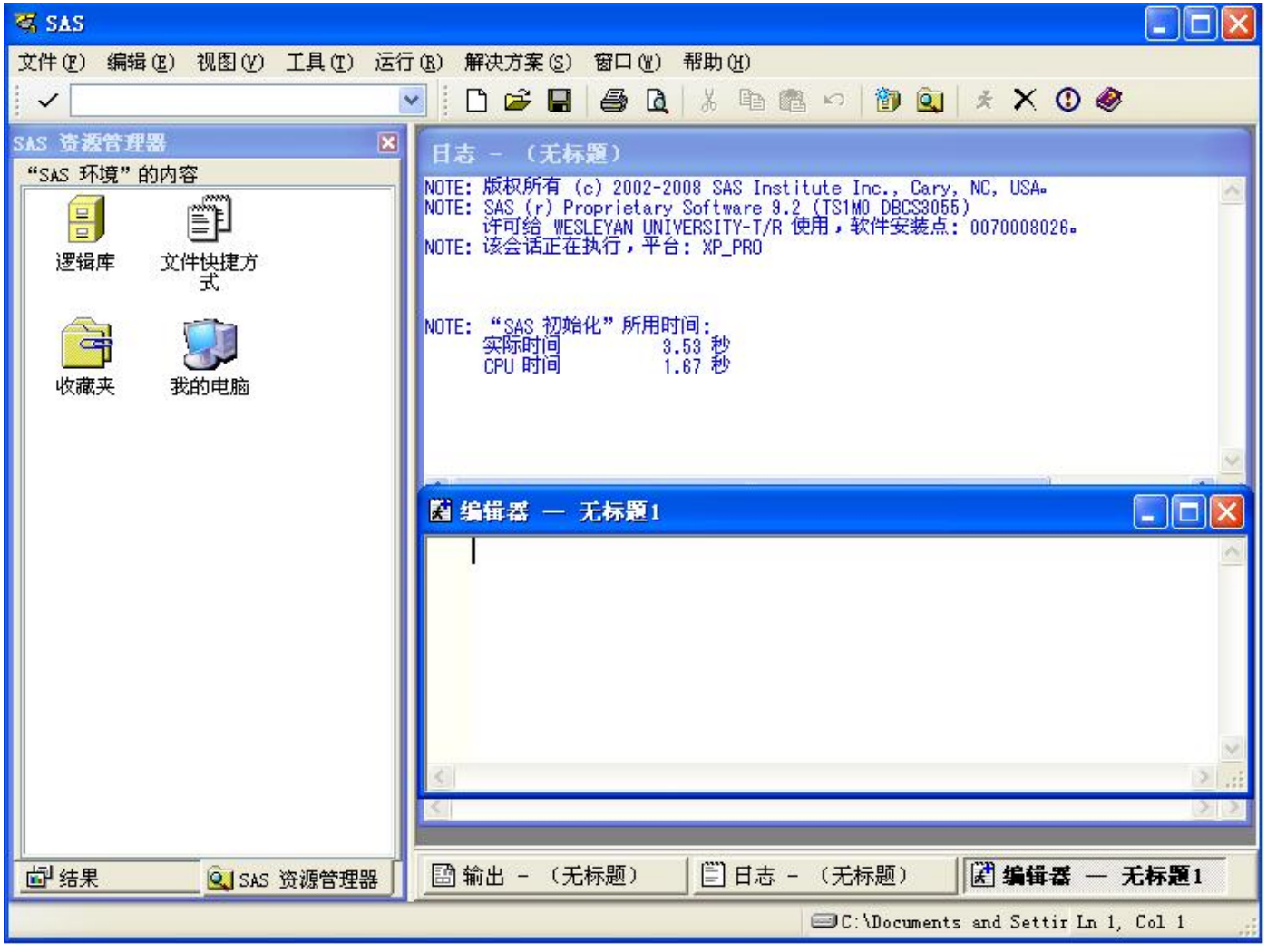
8. 采用输出分发系统

所有的输出都由输出分发系统处理, 它允许用户自己定制显示哪些结果, 输出分发系统可以生成多种格式的结果, 包括HTML文件.

9. 功能强大的系统阅读器

SAS 阅读器是一个浏览SAS各类文件的应用软件. 它提供了一个快速、便利的 查看打印数据集、目录、传施文件、SAS 程序、工作日志、输出结果等内容的通道, 用户 不必运行SAS，甚至不需要在计算机中安装SAS, 即可浏览SAS的各类文件.

SAS启动后界面如下图:



➢ 增强型编辑器(Enhanced Editor): 即程序编辑器. 用于修改或者创建SAS程序. 在命令窗口输入*wpgm*可以激活此窗口. 新版的SAS可以用不同颜色来区分程序的不同部分:

深蓝色	数据步，程序步的开始和结束
蓝色	关键字
棕色	字符串
浅黄底色	数据块
红色	可能的错误

➢ 日志(Log)窗口: LOG窗口用于输出程序在运行时的各种相关信息， 记录执行过的每一条语句. 红色—错误, 兰色—正常, 绿色— 警告.

➢ 结果输出窗口(Output): 它包含由大多数SAS过程产生的输出. 在SAS会话期间相继产生的输出都附加到OUTPUT窗口内容中.

➢ 结果窗口(Result): 帮助用户浏览和管理所提交的SAS程序输出结果 即结果浏览窗口.

➢ 资源管理器(Explorer): 类似于Windows的资源管理器.

Chapter 2
SAS Language

SAS 语句是由SAS 关键字, 变量, 运算符等组成的字符串, **并以分号结束**的语句.

基本的书写规则:

① 语句可以在行的任意列开始和结束;

② 语句的词之间可以用多个空格符或者其他特殊字符隔开.

③ 多个语句也可以写在同一行.

但是,规则的书写有助于阅读和检查, 常用的规则有

④ 不同的程序步和数据步之间留有空行.

⑤ 每个语句都另起一行. PROC和RUN语句由第一列开始书写.

⑥ 语句的词之间都有一个固定的空格符.

例如, 一个规则的SAS程序语句:

↑Example

```
title '95级1班学生成绩排名';
libname mywork 'F:\备课\Stat-Comp\SAScode';
data mywork.c9501;
input name $ 1-6 sex $ math chinese;
avg = math*0.5 + chinese/120*100*0.5;
cards;
  李明   男  92   98
  张红艺 女  89   106
  王思明 男  86   90
  张聪   女  98  109
  刘颖   女  80  110
  ;
run;

proc print;
run;

proc sort data=mywork.c9501;
     by descending avg;
```

↓Example

为了便于了解程序的行数, 可以在命令窗口里输入**nums**来激活程序编辑窗口左边显示行数功能.

Adding comments in SAS

SAS中的注释有两种方式:

- ☞ 注释语句: 使用星号"*"开始, 以分号";"结束. 可占多行.
- ☞ 注释段落: 使用/*开始, 以*/结束. 可占多行.

2.1 Proc Step and Data Step

每个SAS程序, 可以由多个部分构成. 一些简单的程序, 是由许多完成单个动作的程序步 和一些设定环境的语句构成. SAS 只有两种程序步:

- ☞ 数据步(DATA Step): 读入源数据文件和SAS数据集. 修改、编辑或创建 SAS 数据集或文本文件。

☞ 过程步(PROC Step): 面向SAS数据集, 完成特定的计算、分析和呈现的功能。 每个程序步是由若干个语句构成, 每个语句是由一个关键词开始. 并以分号结束、通 常就用开始的关键词命名这个语句, DATA语句和PROC语句分别标志着数据步和过程步 的开始. RUN语句或另一个程序步的开始标志着前一个程序步的结束.

PROC步的一般语法结构为

↑Code

```
PROC 过程名 <data=数据集名> <选项>;
  该过程的专用语句描述;
  <VAR 变量列表>;
  <WHERE 记录选择条件表达式>;
  <BY 变量名>;
RUN;
```

↓Code

常用的过程步有

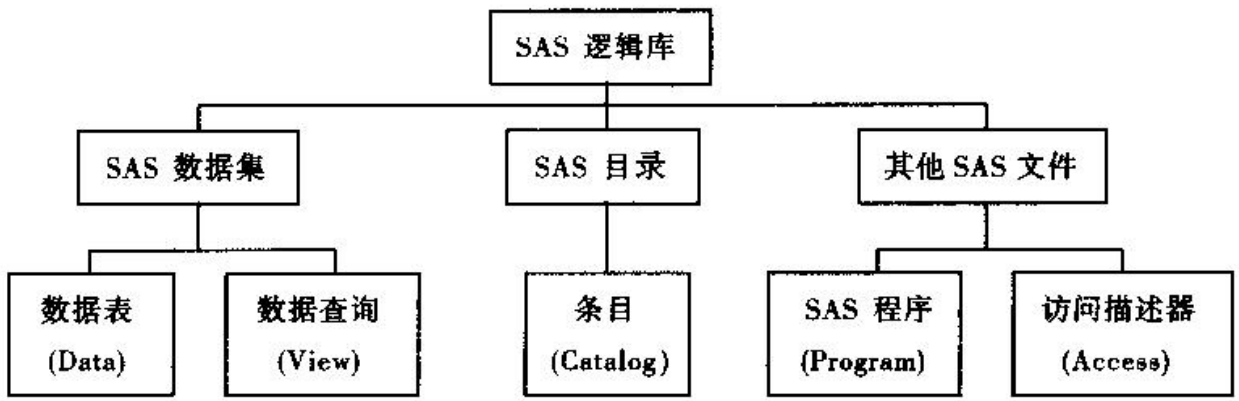
过程步名	描述
SORT	将制定的数据集按指定的变量排序
PRINT	将数据集中的数据列表输出
GCHART	绘制高分辨率的统计图
MEANS	对指定的数据集进行简单的统计描述
UNIVARIATE	对指定的数据集进行详细的统计描述
FREQ	对指定的分类变量进行统计描述和检验
TABULATE	对指定的变量进行制表输出
TTEST	t检验
REG	回归
CORR	相关分析
GLM	广义线性模型
LOGISTIC	逻辑回归分析
NPAR1WAY	非参数检验
ANOVA	方差分析

2.2 SAS Logical Library

SAS的分析和呈现数据过程都只是面向SAS数据集, 它是一类由SAS创建的文件. SAS数据集是其中的一类文件. 在SAS中,不同的文件可以按照不同需要归

入若干 个SAS逻辑库, 以此来对SAS文件进行访问和管理. 一个SAS逻辑库就是一组SAS文件.

SAS逻辑库也是一个逻辑概念. 在Windows环境下, 一个逻辑库就是存放在同一文件夹 或者几个文件夹中的一组SAS文件. 其组成结构如下



创建SAS逻辑库可以在Windows窗口下使用图形界面工具建立, 也可以在SAS程序编辑器中使用命令**libname**来创建:

↑Code

```
libname 逻辑库名 '物理路径';
```

↓Code

例如创建一个物理位置是`c:\mywork`文件夹, 逻辑库名mywork的SAS逻辑库:

↑Example

```
libname mywork 'c:\mywork';
```

↓Example

2.2.1 Access SAS file

在SAS中, 为了访问一个SAS文件, 一定要为该文件所在的位置指定一个SAS逻辑库, 即赋予一个逻辑库名. 逻辑库名的命名规则为

- 不区分大小写,
- 以英文字母或下划线开始
- 由数字字母和下划线构成,总共1-8个字符.

指定好逻辑库名后, 就可以使用两级命令的方式引用文件: 逻辑库.文件名, 比如 sasuser.class是指逻辑库sasuser中名为class的SAS文件. 至于逻辑库sasuser的物理位置, 是通过指定逻辑库名来建立sasuser与某一物理位置的对应.

在每个SAS进程开始时, 系统会缺省的创建名为Work的SAS逻辑库, 其物理位置在 一个临时文件夹中. 当SAS进程结束时, Work中的全部内容都被删除, 因此这是一个 临时逻辑库. 在引用Work 中的SAS文件时, 可以省略逻辑库名.

除逻辑库Work外, 其他逻辑库都是永久逻辑库, 即其中的内容在SAS结束后仍然保留. 其中系统有两个自动指定的永久逻辑库 SASHELP 和SASUSER.

2.2.2 View SAS library and file

在SAS浏览器窗口可以很方便的查看SAS逻辑库,SAS数据集的属性和内容. 在编程 环境下, 也可以通过如下过程步来实现:

- 查看SAS逻辑库的属性和内容

```
PROC datasets lib= 逻辑库名;
RUN;
```

↑Code ↓Code

- 查看SAS数据集的属性和内容

```
PROC contents data= 数据集名;
RUN;
```

↑Code ↓Code

- 输出数据集的数据部分

```
PROC print data= 数据集名;
RUN;
```

↑Code ↓Code

Chapter 3

SAS Programming

3.1 Reading data by DATA STEP

SAS中基本的数据输入语句包括

☞ DATA; 数据步的第一条语句, 同时命名将要创建的SAS数据集名. 一般使用形式为`DATA 数据集名;` 比如

```
DATA sasuser.data1; DATA mywork.data2;
```

↑Example ↓Example

☞ INPUT; (将源数据文件中的数据读入SAS数据集.) 使用形式为

```
INPUT 变量名[$] 起始列-结束列 .....;
```

↑Code

↓Code

其中$ 表示变量为字符型, 其他类型的变量则不需要此符号. 起始列和结束列用以界定变量 取值的位置, 如果不提供, SAS则使用默认值. 例如

↑Example

```
INPUT x y z; INPUT x1-x10; INPUT x $ y @@;
INPUT id 1-2 casid 3-5 n 6 sex 7 age 8-9 edu 10 (v1-v5)(5*1);
```

↓Example

☞ LIST; (列表显示数据的输入格式,以便检查数据是否 输入在它对应的栏位上)

☞ CARDS; (或DATALINES); (告诉SAS读取CARDS后面的数据行) 使用格式为

↑Code

```
CARDS;
数据
;
```

↓Code

CARD和DATALINES功能相同, 区别在于DATALINES只适用于SAS8以后的版本, 而CARDS通用. 再输入数据 包含分号;时, 要使用CARDS4或者DATALINES4并且以4个连续的分号表示结束. 例如

↑Example

```
DATA temp;
INPUT x$ y z;
CARDS4;
24;77 12 23
32;54 23 22
;;;;
RUN;
```

↓Example

☞ INFILE '数据文件名'; (调用外部源数据文件)

↑Example

```
DATA older;
INFILE 'older.dat' /*调用当前目录中的older.dat数据文件*/
INPUT idl—2 casid 3—5 sex 7;
其他数据转换语句;
RUN;
```

↓Example

☞ RUN; (表示数据步结束)

其中DATA和INPUT两条语句在数据读入中是必需的.

3.2 Output format

输出格式语句决定SAS在各种展示结果的报表中如何显示数据.

用Format语句指定输出格式

通常, 可以在数据步的INPUT语句后面加入FORMAT语句, 其一般形式为

↑Code

```
FORMAT 变量名 输出格式;
```

↓Code

例如数据源文件inp.dat里包含如下数据

```
01MAR90LOR198

02MAR90FRA207

03MAx90LON205
```

则使用FORMAT语句使得日期的显示符合习惯(年-月-日):

↑Example

```
DATA mywork.inp;
INFILE 'F:\备课\Stat-Comp\SAScode\inp.dat';
INPUT date date7. dest$3. boarded 3.;
FORMAT date yymmdd10.;
RUN;
```

↓Example

结果在日志窗口.

使用PUT语句改变输出格式

PUT语句可以使得输出按照用户自己的方式输出. 其使用形式为

PUT <<'字符串'> <变量名<=>><对象.属性<=>>>|_ALL;

例如

↑Example

```
DATA mywork.l1;
INPUT id 1-2 sex$ 4 age 5-6 height 8-10 weight 12-14 .1;
PUT id sex age; /*显示3个变量之值*/
PUT id= weight=; /*在等号后面显示变量值*/
PUT '学生代号是:' id '身高=' Height '体重=' weight;
LIST;
CARDS;
01 M19 173 672
02 M20 175 575
03 F19 160 540
04 F20 158 585
;
PROC PRINT;
RUN;
```

↓Example

用label语句定义变量标签

由于SAS的变量名必须以字母为首而且不能超过8个字符, 所 以当一个变量名不足以说明变量的真正含义时, 则用LABEL语句解释变量名. 其格式为:

↑Code
↓Code

```
LABEL 变量名=’标签’ ;
```

其中“变量名”应是INPUT语句中已定义的某个变量名. “标签”则是对变量名的 更详细更明确的注解, 可放在程序DATA语句的后面. 标签内容可以是英文(或汉语拼 音或汉字). 长度在40个字母以内.

例如

↑Example

```
DATA  older;
```

```
LABEL id='地区名称';
casid='观察值编号';
edc='老年人的教育水平';
ocul='退休前的职业';
INFILE 'older.dat';
INPUT id 1-2 casid 3-5 sex 7 age 8-9 edc 10 ocul 11;
...
```

↓Example

用FORMAT过程定义数值标签

计算机所读取的数据一般是数字或字母, 为了让他人能读懂数据(比如1,2,3的真正含义), 通常还应在命令文件或程序中用FORMAT 命令解释数值. 这就是所谓的 数值标签, 其格式如下:

↑Code

```
PROC FORMAT;
    VALUE 格式名   值范围1=' 标签1'
                   值范围2=' 标签2'
                   ...;
RUN;
```

↓Code

例如

↑Example

```
DATA  older;
LABEL id='地区名称';
casid='观察值编号';
edc='老年人的教育水平';
ocul='退休前的职业';
INFILE 'older.dat';
INPUT id 1-2 casid 3-5 sex 7 age 8-9 edc 10 ocul 11;
PROC FORMAT;
   VALUE SEXFMT 1='男' 2='女';
   VALUE OCUFMT 1='工人' 2='干部' 3='商业服务人员' 4='教师'
         5='科技人员';
PROC FREQ;
FORMAT sex SEXFMT. ;
...
```

↓Example

缺失值的表示

如果数据中某个值缺失, 则可以用”.”来表示. 例如

↑Example

```
INPUT edc ocu1 sal1 @@;
CARDS;
09 1 1500 12 2 1600 16 4 1500
12 . 2500 12 . 1300 14 4 1700
...
;
```

↓Example

3.3 Manipulate datasets

3.3.1 SET statement

SET 语句的使用非常灵活, 其可以从已有的SAS数据集中读出指定的变量, 也可以连接和插入数据等等.

↘ 连接(合并)数据. 如果SET语句中有多个数据集名, 则它们就会被(纵向)合并 成一个数据集. 如

↑Example

```
data fitness;
   set health exercise well;
run;
```

↓Example

如果在set语句中的多个数据集中有相同的变量名, 则合并时就会合并相同的变量名, 缺失的数据自动用”.”表示.

↘ 记录的筛选, 下例中数据集nc.memebers里的city为Raleigh的记录被 读取出来到新的数据集realeigh.members.

↑Example

```
data raleigh.members;
   set nc.members;
   if city='Raleigh';
run;
```

↓Example

- 使用keep或则drop语句来筛选变量

↑Example

```
data fitness1;
   set fitness1;
   keep health exercise;
run;
```

```
data fitness1;
   set fitness1;
   drop well;
run;
```

↓Example

- 使用rename语句改变变量的名字

↑Example

```
data fitness1;
   set fitness1;
   rename health=h exercise=ex;
run;
```

↓Example

3.3.2 SORT proc

SORT过程可以对数据集按照指定的变量进行排序, 其语法结构如下

```
PROC SORT <选项>;
 BY <descending> 变量1 <descending> 变量2 ...;
RUN;
```

↑Code ↓Code

其中*descending*只对其后的一个变量起作用, 默认是升序排列. 例如

```
PROC SORT data=mywork.c9501 out=mywork.c9501o;
 BY descending chinese;
RUN;
```

↑Code ↓Code

3.4 Logical statements

3.4.1 IF-THEN statement

语法格式如下

↑Code

```
IF 条件 THEN
   程序块
ELSE
   程序块
```

↓Code

ELSE语句及其后面的程序块可以省略, 如果程序块有多个语句, 则需要以DO开头, END结束. 例如

↑Example

```
INPUT x y @@;
IF X>50 THEN class=1;
ELSE class=2;
CARDS;
```

```
34 56 47 65 87 34 35 68
;
```

↓Example

3.4.2 SELECT-WHEN statement

语法格式如下

↑Code

```
SELECT (表达式);
WHEN (数值1) 执行语句A;
WHEN (数值2) 执行语句B;
...
OTHERWISE 执行语句Z;
END;
```

```
SELECT;
WHEN (条件1) 执行语句A;
WHEN (条件2) 执行语句B;
...
OTHERWISE 执行语句Z;
END;
```

↓Code

例如

↑Example

```
INPUT x y @@;
SELECT;
 WHEN (x<50) class=1;
 WHEN (x>=50) class=2;
 OTHERWISE;
END;
CARDS;
34 56 47 65 87 34 35 68
;
```

↓Example

3.4.3 DO-ENDS statement

语法格式如下

↑Code

```
DO 起始条件 TO 终止条件;
程序块
END;
```

↓Code

例如

↑Example

```
DATA mywork.t2;
DO class=1 TO 2;
   INPUT x y @@;
   OUTPUT;
END;
CARDS;
34 56 47 65 87 34 35 68
;
RUN;
```

↓Example

程序中使用OUTPUT强行将当前数据存为一条记录, 否则程序将在运行到RUN才会生出记录. 两种结果是不同的.

这是由于在默认情况下, 数据步将所读入的数据存放在缓存中, 在依次执行完全部语句后才将相应 信息写成一条新的纪录. 如果希望在一个执行周期中生成多条数据记录, 则需要使用OUTPUT 语句强制输出一条新记录, 如下面的程序:

↑Example

```
DATA mywork.t3;
INPUT x y;
z=x; OUTPUT;
z=y; OUTPUT;
CARDS;
11 22
;
RUN;
```

↓Example

3.4.4 DO-WHILE DO-UNTIL statement

语法格式如下

↑Code

```
DO WHILE 循环满足的条件;
  循环体语句;
END;
```

```
DO UNTIL 循环终止的条件;
  循环体语句;
 END;
```

↓Code

3.5 OPERATIONS

SAS中的运算符号有:

- ☞ 算术运算: 加(+), 减(-), 乘(*), 除(/)和乘方(**).
- ☞ 逻辑判断: 大于(>或GT), 大于等于(>=或GE), 小于(<或LT), 小于等于(<=或LE), 等于(=或EQ), 不等于(^=或者NE), 包含(IN).
- ☞ 逻辑运算: 与(&或AND), 或(|或OR), 非(^或NOT).

Chapter 4
Basic statistical analysis

4.1 Descriptive Statistics Proc

描述性统计指标的计算可以用四种不同的SAS过程来实现, 它们分别是**means**过程、 **summary** 过程、**univariate** 过程以及 **tabulate** 过程。它们在功能范围和具体的操作方法上存在一 定的差别, 下面我们先大概了解一下它们之间的异同.

相同点:

- 它们均可计算均值、标准差、方差、总和、加权和、最大值、最小值、校正的和未校正的离差平方和、变异系数、样本分布位置参数的t检验统计量、缺失数 据和非缺失数据个数等;

- 均可通过by语句将数据分为若干个子数据集，从而对各个数据集 分别进行独立的统计分析.

不同点:

- *means*过程,*summary*过程,*univariate*过程均可以计算 样本的偏度和峰度, 而*tabulate*不计算这些量.
- *univariate*过程可以计算样本众数,其他三个过程不计算.
- *summary*过程执行后不会自动给出分析的结果, 须调用output语句和print语句 来显示分析结果, 而其他三个过程则会自动显示分析的结果.
- *univariate*过程具有统计制图功能, 其他三个过程没有.
- *tabulate*过程不产生输出数据文件, 其他三个过程均可产生输出数据文件.

常用的SAS绘图过程有**chart**过程, **plot**过程, **gchart**过程和**gplot**过程.

4.1.1 MEANS proc

一般形式

```
PROC MEANS 选项 统计值;
     Var 变量名;
     Class 变量名;
     Freq 变量名;
     Weight 变量名;
     Id 变量名;
     By 变量名;
     Output Out='数据集名' '输出统计值' '输出数据集包含的变量名' ...;
RUN;
```

↑Code ↓Code

其中,

选项:

data(要计算的输入数据集的名字),
NOPRINT(选择不输出任何过程处理后的统计值),

PRINT(选择输出处理后的统计值，为缺省选择)，
VARDEF=DF|WEIGHT|WGT|N|WDF| (在方差的计算中设定除数).

统计值:

默认计算平均值，标准差，最大值和最小值.
否则将根据列出的'统计值'进行计算,可取的值有mean, std,
stderr(平均值的标准差),min, max, sum,
USS(每个变量原始值的平方和),CSS(对平均值修正的平方和),
CV(变异系数), SKEWNESS, KURTOSIS,var,
sumwgt(WEIGHT变量值的和)等等。

例如考虑一组26个摩托车信息, 每个记录包括生产商(MAKE), 价格(PRICE), mpg(MPG), 维修记录 (REP78), 国产还是进口(FOREIGN). 数据文件为auto.txt. 读入数据的SAS代码如下

↑Example

```
DATA mywork.auto;
INFILE  'F:\备课\Stat-Comp\SAScode\auto.txt';
INPUT  MAKE $ PRICE MPG REP78 FOREIGN;
RUN;
PROC PRINT DATA=mywork.auto(obs=10);
RUN;
```

↓Example

使用*means*过程分析数据

↑Example

```
PROC MEANS DATA=mywork.auto;
  CLASS foreign ;
  VAR mpg;
RUN;
```

↓Example

4.1.2 SUMMARY proc

summary 过程的功能与means过程大同小异, 主要执行数据汇总的功能, 同样可对全部 观测或在指定的分组内对指定的变量计算各种指定的描述性统计量, 以及对基于样本对均值 执行t检验。在默认情况下, summary 过程不显示分析 的结果, 需要通过特定的语句或选项的控制方式实现分析结果的显示或输出. 其语法结构和means 过程完全相同.

例如

↑Example

```
PROC SUMMARY print DATA=mywork.auto;
  CLASS foreign ;
  VAR mpg;
RUN;
```

↓Example

4.1.3 UNIVARIATE proc

univariate 过程的主要功能是进行数据汇总和数值型变量分布情况的描述, 还可以绘制高分 辨率的用于描述数值型变量分布情况等的统计图形, univariate过程也可基于样本进行均值 的t检验. univariate 过程可实现的功能简要归纳为如下的列表:

- 计算以矩为基础的描述性统计量;
- 计算中位数、众数、Range、以及各种分位数;

➡ 对位置参数(location)和尺度参数(scale)进行稳健估计;

➡ 计算置信区间;

➡ 给出极值及其所对应观测的列表;

➡ 创建有关数据的频数表;

➡ 绘制有关数据分布情况的统计图形;

➡ 执行有关分布位置和正态性的检验;

➡ 执行拟合优度检验的操作;

➡ 绘制直方图，还可为所拟合的连续性分布选择性添加分析密度曲线;

➡ 针对多种理论分布绘制Q-Q图及概率图，并可为其添加与有关位置参数或尺度参数 (自定义值或估计值)对应的参考线;

➡ 绘制单因素或两因素的对比直方图、对比Q－Q图或对比概率图;

➡ 为所绘制的高分辨率统计图添加包含有关统计量的插页列表框;

➡ 创建包含指定统计量或有关拟合分布参数估汁值的输出数据集。

其语法结构为

↑Code

```
PROC UNIVARIATE 选项;
 BY 变量;
CLASS 变量1 <(v-options)> < 变量2 <(v-options)> >
           < / KEYLEVEL= value1 | ( value1 value2 ) >;
FREQ 变量;
HISTOGRAM  变量/选项;
ID 变量;
INSET keyword-list < / options > ;
OUTPUT < OUT=SAS-data-set >
       < keyword1=names...keywordk=names > < percentile-options >;
PROBPLOT 变量/选项;
QQPLOT  变量/选项;
VAR 变量 ;
WEIGHT 变量;
```

↓Code

例如

↑Example

```
PROC UNIVARIATE DATA=mywork.auto;
```

```
  VAR mpg;
  HISTOGRAM mpg / cframe = ligr
                  cfill  = blue;
;
RUN;
```

↓Example

另外一个例子

↑Example

```
data Plates;
      label Gap = 'Plate Gap in cm';
      input Gap @@;
      datalines;
   0.746  0.357  0.376  0.327  0.485 1.741  0.241  0.777  0.768  0.409
   0.252  0.512  0.534  1.656  0.742 0.378  0.714  1.121  0.597  0.231
   0.541  0.805  0.682  0.418  0.506 0.501  0.247  0.922  0.880  0.344
   0.519  1.302  0.275  0.601  0.388 0.450  0.845  0.319  0.486  0.529
   1.547  0.690  0.676  0.314  0.736 0.643  0.483  0.352  0.636  1.080
   ;
   run;

   title 'Distribution of Plate Gaps';
```

```
ods select ParameterEstimates GoodnessOfFit FitQuantiles MyHist;
proc univariate data=Plates;
   var Gap;
   histogram / midpoints=0.2 to 1.8 by 0.2
               lognormal
               weibull
               gamma
               vaxis   = axis1
               name    = 'MyHist';
   inset n mean(5.3) std='Std Dev'(5.3) skewness(5.3)
          / pos = ne  header = 'Summary Statistics';
   axis1 label=(a=90 r=0);
run;
```

↓Example

4.1.4 TABULATE PROC

tabulate 过程以表格的形式呈现数据部分或全部变量的各种描述性统计量, 是常用的 报表制作工具之一，我们用它可以制作各种各样的简单或高度复杂的表格。

其语法结构为

↑Code

```
PROC TABULATE data=数据集名;
 BY 变量;
CLASS 变量;
FREQ 变量;
KEYLABEL 关键词;
TABLE  页维说明,行维说明,列维说明/选项;
VAR 分析变量 ;
WEIGHT 变量;
RUN;
```

↓Code

例

↑Example

```
proc tabulate data=mywork.c9501;
  class sex;
  var math chinese;
  table sex, (math chinese)*(mean std);
run;
proc tabulate data=mywork.c9501;
```

```
  class sex;
  var math chinese;
  table  (sex all)*(N PCTN);
run;
proc tabulate data=mywork.c9501;
   class sex;
   var math chinese;
   table (sex all), (math chinese)*(mean std);
   keylabel mean='平均值' std='标准差' all='总计';
   label sex='性别' math='数学' chinese='语文';
run;
```

↓Example

4.1.5 GCHART proc

GCHART 过程可以绘制6种图形:方块图, 水平或垂直的柱状图, 饼图和圆环图, 以及星图. 其语法 结构为

↑Code

```
PROC GCHART [ DATA=<数据集名> [选项] ] ;
     HBAR <变量名列> / [选项] ;  *绘出条形图;
```

```
    VBAR <变量名列> / [选项];  *绘出水平条形图;
    BLOCK <变量名列> / [选项]; *绘出三维直方图;
    PIE <变量名列> / [选项]; *绘出饼图;
    STAR <变量名列> / [选项]; *绘出星状图;
    AXISn [选项]; *控制坐标轴的形状和颜色;
    BY <变量名列>; 按该变量取值分层绘制，要求数据集已按该变量排序
RUN;
```

↓Code

例如

↑Example

```
data totals;
length dept $ 7 site $ 8;
input dept site quarter sales;
datalines;
Parts Sydney 1 7043.97
Parts Atlanta 1 8225.26
Parts Paris 1 5543.97
Tools Sydney 4 1775.74
Tools Atlanta 4 3424.19
Tools Paris 4 6914.25
;
```

```
title "Total Sales";
footnote j=r "GCHBKSUM ";
proc gchart data=totals;
format sales dollar8.;
block site / sumvar=sales;
run;

proc gchart data=totals;
format sales dollar8.;
hbar site / sumvar=sales;
run;
```

↓Example

4.1.6 GPLOT proc

GPLOT 可以绘制两种散点图: 气泡形状以及点状. 语法结构如下

↑Code

```
PROC GPLOT [ DATA=<数据集名> [选项] ] ;
    BUBBLE 纵坐标变量*横坐标变量=气泡尺寸变量名 / [选项];
    BUBBLE2 纵坐标变量*横坐标变量=气泡尺寸变量名 / [选项];
```

```
    PLOT  <纵坐标变量*横坐标变量[=分层变量名]...> / [选项];
    PLOT2 <纵坐标变量*横坐标变量[=分层变量名]...> / [选项];
       *在原图基础上重叠绘制第二幅散点图;
    SYMBOLn [选项]
       *定义符号、添加趋势线、定义点和线的颜色;
  BY <变量名列>;
```

↓Code

↑Example

```
proc gplot data=sashelp.shoes;
     where Region
     in("United States","Eastern Europe");
     plot Returns * Sales = Region;
     bubble Returns * Sales =Stores;
run;
```

↓Example

4.2 INFERENTIAL Statistics

4.2.1 T-TEST

例如考虑一组汽车数据, 其中对AMC汽车的MPG数值缺失. 读入此数据

↑Example

```
DATA mywork.car;
  INFILE 'F:\备课\Stat-Comp\SAScode\car.txt';
  LENGTH make $ 20 ;
  INPUT make $ 1-17 price mpg rep78 hdroom trunk weight
        length turn displ gratio foreign ;
RUN;
```

↓Example

使用t检验来检验进口和国产车是否在MPG上有差异.

↑Example

```
PROC TTEST DATA=mywork.auto;
  CLASS foreign;
  VAR mpg;
RUN;
```

↓Example

4.2.2 Chi-square tests

使用卡方检验检验维修记录和生产地之间是否独立

↑Example

```
PROC FREQ DATA=mywork.car;
  TABLES rep78*foreign / CHISQ ;
RUN;
PROC FREQ DATA=mywork.car;
  TABLES rep78*foreign / CHISQ EXACT ;
RUN;
```

↓Example

4.2.3 Correlation

使用CORR过程来检查price, mpg和weight之间的相关性:

↑Example

```
PROC CORR DATA=mywork.car;
  VAR price mpg weight ;
RUN;
```

↓Example

4.2.4 Regression

使用回归过程来对price,mpg和weight建模:

↑Example

```
PROC REG DATA=mywork.car;
  MODEL price = mpg weight ;
RUN;
```

↓Example