《计量软件在金融中应用》

习题集

沈根祥

(上海财经大学经济学院)

2021.8

第一部分：习题

第一章

1． 通过SAS界面中菜单栏“帮助”→“SAS网站”→“客户支持中心”注册SAS个人账户（SAS Profile）。登陆（Sign in）SAS客户支持中心（SAS Customer Support）页面，点击“Connect”进入SAS社区（Communities）或博客（Blogs）。熟悉SAS客户支持系统，寻找自己感兴趣的社区和博客。点击“Communities”→“ Find a Community”选择一个社区进入（例如SAS Programming），“Pick your Board & Ask a Question”提出一个问题。或者在该社区条目下选择一个小组（Board）进入（例如SAS Programming），通过“New Message”提出问题。

2． 什么是SAS的当前窗口？如何实现当前窗口切换？在多次操作中，如果某个SAS窗口不见了，如何调出该窗口？

3．SAS资源管理器中的“文件快捷方式”有什么功能？建立一个SAS程序文件的快捷方式。

4．工具栏中很多功能与菜单栏中菜单功能相同。找出菜单栏菜单功能中没有的工具栏功能。

5． 在SAS资源管理器中通过“收藏夹”→“我的桌面”，对Windows桌面的文件进行操作（复制、粘贴、打开等）； 通过“计算机”→“本地磁盘（D:）”在磁盘D:根目录下建立目录“mydata”。

第二章

1.用Proc contents过程查看数据集sasuser.airports的描述信息。写出所有者名、文件路径和变量city的标签。

2.启动SAS后通过lib命令查看临时逻辑库work对应的物理目录路径。关闭SAS后还能找到work对应的物理目录吗？为什么？再次启动SAS，查看work对应的物理目录，和上次一样吗？为什么？

3.建立逻辑库sasdata，对应的物理目录为D:\Lecture\SAS\mydata。为什么SAS不允许在逻辑库内建立子逻辑库？

4.在逻辑库中选择一个数据集，鼠标右键点击，弹出菜单中选择“重命名”，在对话框中输入汉字“文件”，确定，会提示什么信息？为什么？可以给数据集设置标签吗？

5.“万能”日期输入格式ANYDTDTE .将外部数据02/03/12读作什么？用什么输入格式才能分别读作2012年2月3日、2012年3月2日？有将其读为2002年12月3日的输入格式吗？

6.用什么样的输入格式才能将会计金额形式的外部数据($7.8)读为-7.8？

7.将sashelp.class输出为Excel 4表格（Microsoft Excel 4 Spreadsheet），class.xls，然后用“导入数据”将其导入为SAS数据集，在“select a data source from the list”下拉菜单中选择“Microsoft Excel Workbook”，将会出现什么问题？如何解决？如果在给出导入的SAS数据集名字时，输入“123”会出现什么问题？为什么？

第三章

表达式3+’2’、3||’2’正确吗？可行吗？表达式3+’a’正确吗？可行吗？

'2018/05/03'd 是正确的日期常量表达吗？'3Feb19'd 是正确的日期常量表达吗？SA 将其中的'19'识别为世纪还是年份？

2.如果 DATA 步中出现赋值语句 y=sim(x);，信息窗口会出现什么错误提示？SAS 将 sim(x)识别为一个变量还是函数？为什么？

3.通过查阅帮助,详细说明函数 logistic()、fact()、substr()=、exist()和 Yrdif()的功能和使用方法。

4.将 SAS 数据集 sashelp.class 中 name 以'罗'开头的观测挑选出来,if 语句如何写？if name='罗';可以吗？where 语句如何写？还有其它方法吗？语句 where name not like '%德';选出的是哪些观测？

5.什么是空语句？空语句能作为 DATA 步的结束边界吗？

6.DATA 步中的自动变量_Error_和_N_的初次赋值（_Error=0_,_N_=1）是在编译阶段还是在执行阶段？你能编程序验证码？

7.什么是 DATA 步中的声明性语和执行性语句？举出三个声明性语句？

8.指出 DATA 步程序

```
Data test;
set sashelp.class(keep=name sex);
h=height*0.025;
run;
```

中存在的错误并进行修改。指出 DATA 步程序

```
Data test;
set sashelp.class;
h=height*0.025;
where h<10;
run;
```

中存在的错误并进行修改。

9.程序

```
data s;
if sex='男';
set sashelp.class;
h=height*0.025;
run;
```

存在语法错误吗？将取子集 if 语句置于 set 语句之前和之后得到的结果有什么区别？为什么？

10.如何用 input 函数将字符'03APR2019'转换为日期常量？能将'03APR2019'd 转为日期常量吗？如何将'03APR2019'd 转换为字符常量？

11.写出 DATA 步程序用数据集 work.test

VIEWTABLE: Work.Test

	city	year
1	上海	2017
2	上海	2018
3	上海	2019

生成数据集 work.test1

VIEWTABLE: Work.Test1

	city	year1	year2	year3
1	上海	2017	2018	2019

12.在程序不同位置添加语句 put _all_可以查看程序执行不同阶段 PDV 中变量取值的变化。运行程序

```
data test;
put _all_;
set sashelp.class;
```

```
    put _all_;
    run;
```

查看信息窗口的内容，解释

(1)每条观测都出现两次，两次的内容有什么不同？

(2)为什么最后一条观测的第二次显示中，_N_的值是 20？DATA 步内置循环是 20 次吗？如果是 20 次为什么只有 19 条记录输出？你能据此判断在从本次循环到下次循环的过程中，_N_的值是在哪个环节增加的吗？如果将程序改为

```
    data test;
    put _all_;
    set sashelp.class;
    _n_=3;
    put _all_;
    run;
```

怎么解释信息窗口看到的_N_值的变化？

13.提交并运行程序

```
    data _null_;
    set sashelp.class;
    rename height=h;
    keep name age sex;
    run;
```

信息窗口为什么没有出现运行程序 3-31 的警告（warning）信息？

14.用 output 语句在数据集 sashelp.class 第一条观测前添加一条空白记录。用 output 语句和 if-then-语句在第 5 条观测后添加一条空白记录（提示：可使用自动变量_n_或者 set 语句选项 curobs=）。

15.文件 sashelp.class 中有 19 条观测，对应一个班级中 19 位学生的信息，其中变量 age 表示学生年龄。针对每位学生，计算班级中年龄大于该生的学生人数。提示：采用 do-end 循环语句、output 语句、rename=()数据集选项以及带 point=选项和 nobs=选项的 set 语句等。

16.用 output 语句和 DO 循环语句在数据集 sashelp.class 第 5 条观测前添加一条空白记录生成数据集 work.c_5。为什么 work.c_5 中变量 i 的值从第 7 条记录开始为缺失值？

17.为理解 stop 语句和 set 语句 point=选项，运行程序

```
    data x;
    do i=1 to 25;
    set sashelp.class point=i;
    output;
    end;
    stop;
    run;
```

（1） 如果不带 stop 语句将会如何？

（2） 为什么数据集 x 的第 19 到第 25 条观测完全相同？

18. 运行程序

```
    data x1;
    n=23;
    set sashelp.class point=n;
    stop;
    run;
```

为什么数据集 x1 中没有观测？如果在 set 语句添加语句 output 会得出什么步同结果？为什么？

19.执行程序

```
    data test;
```

```
if 0 then
do;
set sashelp.class(drop=name sex);
retain sex name;
end;
set sashelp.class;
run;
```

（1） test 数据集与 sashelp.class 由什么区别？

（2） 程序中的 do-end 程序块中的语句执行了吗？这些语句的作用是什么？

（3） 为什么需要第二个 set 语句？

20.对程序 3-66 进行修改，在数据集 work.test 中每只股票的最后一条观测后和下一只股票第一条观测前添加一条空白记录。

21.将数据集 rate 和 GDP、cpi 合并，要求

（1）从 cpi 中抽取 3/6/9/12 月份的观测作为季度观测值；

（2）从 rate 中抽取每个月份最后一个交易日数据作为月份交易数据，再从月份数据中抽取 3/6/9/12 月份的观测作为季度观测值

（3）实施合并。

22.将 Animal 和 Plant 进行非匹配并接合并（merge），将结果与程序-70 的合并结果比较，据此说明两种并接合并的区别。

23.将数据集 c_Num 变量 No 的最后一个观测值和 sashelp.class 的所有观测匹配合并。

24.将程序 3-75 中的 if 语句的位置改变为

```
data class_num2;
set sashelp.class;
set c_num point=_n_;
if name=:'罗';
run;
```

会得出什么结果，为什么？

第四章

1．执行 data 步程序

```
data jansales;
input item  $10. amount;
datalines;
trucks 1382
vans 1235
sedans 2391
   ;
run;
```

能够正确读取内嵌数据行吗？为什么？如何修改？（提示：采用格式修饰符:）.

2．如何用 input 语句将内嵌数据

```
datalines;
054 性别=男 姓名=张三
069 性别=女 姓名=李英
 ;
```

读给三个字符变量 id、sex 和 name？写出 data 步程序代码。

3.将程序 4-12 中变量 region 的读取方式改为列表方式，如何设置列相对指针控制符才能正确读取 date 变量和其余变的值？

4．文本文件 CS300_2017.txt 中为沪深 300 样本股（2017 年）的股票代码、股票名称等信息。

（1）用 data 步 input 语句将股票代码（去掉其中的 ZH 和 SZ）其读入 PDV 变量 STKCD，再用 put 语句将引号引起来的股票代码读出（不用生成新的数据集）到文本文件 temp。

（2）用 put 语句在 temp 中形成如下文本

```
Where stkcd in (list);
```

list 是沪深 300 样本股票代码列表，股票代码用摇号引起来，股票代码之间逗号分隔。形如

```
Where stkcd in ('601398','600028','601988');
```

5.为什么程序 4-45 中需要 Stop 语句，而程序 4-46 中却不需要？

6.用 infile 语句加上选项 memvar=vname 并为 vname 赋值空格，结合并 input 语句生成数据集 A，A 中变量 name 的观测值为目录 d:\mydata 下所有文件名。这为获取电脑某个目录下的所有文件名提供了方便。

7.运行以下DATA步

```
data s;
x=2;y=3;
set sashelp.class;
array nc[*] _numeric_;
put nc[5];
run;
```

对信息窗口输出内容进行解释。

8.将程序 4-65 中生成的数据集 want（面板数据形式）还原为原来的数据摆放形式。

9.修改程序4-66，使计算5日移动平均时包括当日。

10. 为什么程序 4-70 中 set 语句采用选项 point=但后续并没有 stop 语句，DATA 没有进入死循环？

11.修改程序 4-70，计算（每位学生）名字排在其后面的学生中年龄大于该生的学生人数。

12.计算 sashelp.class 中男生和女生身高的中位数（median）。（提示：先将 sashelp.class 中 sex 变量值'男'和'女'分别修改为'M'和'F'并按 sex 和 height 排序，用 data 步计算男生和女生人数，用例程 sysputx()和函数 symget()在 data 步之间进行数据传递）。

13.采用定义数组的方法重做第 12 题。

14.SAS 数据集 p2p 中有 10 万条观测，为某 p2p 平台上的债权买卖记录。变量 createtime 为交易发生时间，sellid 和 buyid 为卖出者和购买者代码，id 为债权代码。查询一个债券卖出者（sellid）在其卖出日期（createtime）后的 5 天、7 天、10 天、15 天及 30 天内是否有再买债权（buyid），并统计此时买的债权的 id。将查询结果输出 SAS 数据集。

第五章

1. where 数据集选项能够用于 proc 步的输出数据集吗？

2. 用 proc print 分别打印 sashelp.class 中男性观测和女性观测，写出采用 where 语句、where 数据集选项、by 语句的程序。这些程序的执行有什么区别？

3. 什么是 run-组？所有 proc 程序都是交互运行方式吗？如果不用 quit 语句退出，交互执行的 proc 会出现什么问题？

4. 数据集 Fishdata 是在两个池塘 Cole pond 和 Eagle lake 中连续三天捕鱼数据，变量 location 表示池塘，Length1 和 Weight1 表示捕到第一条鱼的重量和长度，其它变量以此类推。数据集的数据结构为

Location	Date	Length1	Weight1
Cole Pond	02JUN95	31	0.25
Cole Pond	03JUL95	33	0.32
Cole Pond	04AUG95	29	0.23
Eagle Lake	02JUN95	32	0.35
Eagle Lake	03JUL95	30	0.2
Eagle Lake	04AUG95	33	0.3

用 proc transpose 对 Fishdata 进行转置，要求转置后形数据集具有如下结构

Location	Date	以前的变量名	Measurement
Cole Pond	02JUN95	Length1	31
Cole Pond	02JUN95	Length2	32
Cole Pond	02JUN95	Length3	32
Eagle Lake	02JUN95	Length1	32
Eagle Lake	02JUN95	Length2	32
Eagle Lake	02JUN95	Length3	33

其中 measurement 变量为鱼的长度。(提示:用 var 语句和 by 语句,在 proc transpose 语句选项 out=给出的输出数据集后采用数据集选项 rename=修改变量名)

5．SAS 数据集 zgsh 中的数据是 2018 年 4 月 2 日到 4 月 10 日的日内 1 分钟高频交易数据，tdate、ttime 和 p 分别为交易日期、交易时间和成交价格。

（1）等距抽样形成 5 分钟、10 分钟、30 分钟、1 小时、日交易价格子数据集。

（2）用 proc univariate 画出 1、5、10、30 分钟、1 小时、日交易价格的频率直方图，同时画出正态分布拟合和核估计拟合的密度曲线图。

（3）能否认为频率越低的交易价格越接近正态分布？用分布检验验证你的观测。

6． 在 data 步中调用例程 execute()，打印（print）程序 5-32 中生成的 SAS 文件 economics、finance 和 math。

7．在 data 步中调用例程 execute()，用 proc means 计算程序 5-32 中生成的 SAS 文件 economics、finance 和 math 中变量 code 的最大值，并输出为同名 SAS 数据集。

第六章

1． 用option yearcutoff=语句将默认世纪跨度设定为1999-2098。然后运行程序

```
data _null_;
a='01MAR15'd;
b='01MAR97'd;
put 'a=' a date10. 'b=' b date10.;
run;
```

日志窗口中将输出什么结果？为什么？

2． 如何正确读取只有年份的外部日期数据？

3． 用函数nwkdom()计算2028年6月的第3周星期6对应的日期。

4． 数据集trt_600085_1m是股票同仁堂（60085）2021.3.22到2021.4.9的日内1分钟高频交易数据。_col2和_col3分别为交易日期和交易时间。

1）采用函数hour()、minute()和second()从变量_col3中差分出时 、分和秒，生成变量h、m和s；

2）用函数dhms()将col2和h、m、s合并为交易日期和时间变量datetime。

3）用函数put()把_col2和_col3转换为字符变量后，用cats函数将其连接为日期时间形式的字符，再用函数input()将其转换为日期变量。

5． 用函数yrdif()计算自己的年龄。

6． 'year2.15'是正确的时间间隔格式吗？你如何判断？'year.13'呢？

7． 函数intnx('week.3','02apr21'd,3)的返回值是什么？

8． 设变量date为日期变量，用函数intnx()将日期按星期1对齐。写出包含自变量的完

整函数形式。

9. 为什么函数intnx('week.3','**01jun20'd**,2)的返回值不是2020年6月16日，而是2020年6月9日？
10. data步滞后函数和通常意义上的滞后函数有什么不同？需要注意些什么？
11. 运行程序3-66生成数据集test中包含三支股票的一周内的收盘价数据。对变量clp取对数然后差分计算日收益。不采用数据集排序以及由此产生的临时变量first.stkcd，将跨越不同股票处的无效收益数据删除。
12. 将数据集rdr7中变量dr7的频率由日降为月，月内最后一天数据作为月度数据。写出data程序代码。
13. 数据集shd1519中变量clp为上证日收盘指数。计算收盘指数的5日平均、120日平均和180日平均，生成变量ma5、ma120和ma180。写出data步程序代码。
14. 用proc sgplot在同一坐标内画出13题生成的ma5、ma120和ma180的时序图，数据图标分别为空心五角星（star）、井号（hash）和实心三角形（trianglefilled），大小为2个像素，颜色分别为蓝色、黑色和红色；折线线型分别为实线（solid）、22号线型和点线（dot）。写出程序代码。
15. 数据集shd1519中变量clp为上证日收盘指数。计算clp的指数平滑，平滑参数λ分别取0.96和0.75，生成变量em96和em75。用proc sgplot在同一坐标内画出em96和em75的时序折线图。写出程序代码。

第七章

.SAS数据集tongrt是股票同仁堂（600085）2016年1月4日至2020年12月31日的日交易数据，变量clp_trt为日收盘价。用proc arima中的identify对变量clp_trt进行单位根检验，采用ADF检验方法，检验模型采用5阶滞后。

2.SAS数据集sh_bank是招商银行（60036）、农业银行（601288）、工商银行（601398）等多家上市银行2016年1月4日至2020年12月31日的股票日交易数据，stkcd、tdate、clp和vol分别表示股票代码、交易日期、收盘价和交易量。

（1）采用where语句对工商银行（601398）的日收盘价进行Phillips-Perron单位根检验；当检验模型采用不同滞后阶数时，检验结果是否发生变化？

（2）首先根据日收盘价计算日对数收益率，然后采用where数据集选项对招商银行的日对数收益率进行ADF单位根检验。当检验模型采用不同滞后阶数时，检验结果是否发生变化？

3.用ESACF确定数据集tongrt中变量clp_trt的ARMA模型的阶数。确定结果是否唯一?你更愿意选择哪个结果建立模型？为什么？如果要建立AR模型，如何采用ESACF确定阶数？确定的阶数是否比ARMA模型中AR阶数高？能够从理论上进行分析？

4.对数据集tongrt中的变量clp_trt建立ARMA模型。如果clp_trt有单位根而其一阶差分Δclp_trt没有单位根，则对Δclp_trt建立模型。

（1）采用MINIC方法确定模型阶数，采用极大似然方法估计模型回归系数。

（2）从数据结果给出的图形，判断残差是否服从正态分布。如果残差不服从正态分布，估计结果可信吗？哪些方面不可信？为什么？

（3）estamat语句中的两个选项outmodel=和outstat=，其功能有什么区别？用这两个选项将clp_trt的ARMA模型估计结果保存为SAS数据集。

（4）如果建立的是Δclp_trt的ARMA模型，用forecast语句进行10步预测，并将预测结果保存为SAS数据集fcast。这样的预测的对Δclp_trt的预测还是对原始变量clp_trt的预测？

5.过程proc autoreg和proc arima的功能有什么区别？如何用proc autoreg过程实现proc arma的功能？

6.SAS数据集sh_yiyao为同仁堂（600085）、太极集团（600129）、复兴医药（600196）等多家医药上市公司2016年1月4日至2020年12月31日的股票日交易数据，stkcd、tdate、clp和vol分别表示股票代码、交易日期、收盘价和交易量。对复兴医药的日收益建立模型。

（1）根据日收盘价计算日收益率变量；

（2）用data步及其where数据集选项生成只包含复兴医药（600196）数据的数据集fxyy；

（3）对日收盘价和日收益率数据分别进行KPSS单位根检验。KPSS检验和其它单位根检验的最大区别是什么？

（4）用proc arima过程确定日收益变量的AR模型阶数；

（5）用proc autoreg过程估计收益率变量的AR模型。

（6）对交易量vol取对数生成变量lgvol。用proc autoreg过程建立收益率变量的AR模型和GARCH（1,1）模型，并在GARCH（1,1）模型中添加外生变量lgvol，并将lgvol在GARCH模型中的系数限制为非负。

7．SAS数据集shindex和bys分别是上证综合指数（83_000001）和白云山（600332）2016年1月4日至2020年12月31日的日收盘信息，clp为收盘指数和日收盘价。为求出白云山股票的贝塔系数，进行如下回归

$$r_{b,t} = c + \phi_1 r_{b,t-1} + \phi_2 r_{b,t-2} + \beta_1 r_{shind,t} + \beta_2 r_{shind,t-2} + \epsilon_t,$$

其中误差项条件方差服从EGARCH（1,1）。

（1）用data步rename命令将shindex和bys中变量clp分别改名为clp_shindex和clp_bys；

（2）用proc sort对两个数据集按tdate排序；

（3）用merge和by语句匹配并接两个数据集，生成新数据集shid_bys；

（4）生成上证指数日收益率变量r_{shind}和白云山日收益变量r_b；

（5）生成r_b的1阶和2节滞后变量，生成r_{shind}1阶和2节滞后变量；

（6）用model语句设定模型，分别用选项nlag=和不使用该选项两种方式；

（7）将EGARCH模型中表示杠杆效应的参数约束为0；

（8）对杠杆效应参数是否为0进行参数约束检验；

（9）对标准化残差进行独立性游程检验，以判断模型设定是否充分。

（10）对模型进行Bai-Perron变点检验，变点个数设定为3。对检验结果进行说明。

第八章

1．对数据集sh_bys_zgsh_r中的变量r_sh、r_b和r_z建立三元VAR模型，写出程序代码。

（1）分别用信息准则AIC、AICC、HQC和SBC进行模型阶数选择；

（2）模型中包含线性时间趋势项；

（3）采用条件极大似然估计估计方法估计模型参数；

（4）通过对估计残差的混成白噪声检验对模型进行检测；

（5）同test语句检验是模型是否应该带线性时间趋势项。

2．数据集sh_bank包含了多家上海证券交易所上市交易的国有股份制银行股票的日交易信息，其中clp为股票日收盘价。

（1）用data步生成建设银行（601939）的数据集bank_js，对clp取对数生成变量lnclp，删除缺失数据；

（2）用语句model建立clp的AR(3)模型，计算clp的分数单整阶数；对lnclp建立AR(3)

模型，计算lnclp的分数单整阶数。比较两个变量的单整阶数能得出什么结论？

（3）确定clp的单整阶数后，用proc expand对其进行分数差分得出变量FDclp。用proc arima对FDclp进行单位根检验。

3．数据集sh_yiyao包含了多家上海证券交易所上市交易的医药类企业股票的的日交易信息，其中clp为股票日收盘价。

（1）用data步生成浙江医药（600216）、羚锐制药（600285）、片仔癀（600436）、和新华医疗（600587）的包含交易日期，收盘价的数据集；

（2）对各个数据集按tedate进行排序，并用data步merge语句合并这些数据集。为了避免相同变量覆盖，请提前将各数据集中的变量clp改名；

（3）用proc sgplot在同一坐标内画出四只股票收盘价时序图；

（4）计算每只股票日收益率，并检验其平稳性；

（5）对四只股票收益率建立VAR(2)模型，用无条件OLS方法估计模型参数；

（6）画出累计脉冲响应图和正交脉冲响应图；

（7）给出5期方差分解结果。

（8）检验股票浙江医药和股票新华医疗日收益是否股票羚锐制药和股票片仔癀日收益的格兰杰原因。

4．数据集shindex中变量clp为上海证券市场的上证收盘指数。

(1) 将shindex和第3题中合并得到的股票收盘价数据按交易日期tdate进行匹配合并（merge）。

(2) 对四只股票收盘价和上证收盘指数进行单位根检验；

(3) 选择一阶单整的股票收盘价和上证收盘指数进行协整检验；

(4) 挑选和上证收盘指数存在修正关系的股票收盘价，建立二者的误差修正模型；

(5) 挑选一阶单整的三只股票收盘价进行协整分析，建立三元误差修正模型。

(6) 将误差修正模型中协整系数和调整系数不显著的元素约束为0，重新估计模型；

(7) 用test语句检验(6)中的约束条件是否成立。

(8) 将最终得到的误差修正项定义为变量，画出其时序图。能从中看出什么？

5．数据集sh_security中包含了申万宏源（000166）、中信证券（600030）等多家上市券商的日股票交易数据，clp为股票收盘价。

(1) 从中挑选海通证券（600837）的日交易数据形成数据集hts；从sh_yiyao中挑选新华医药的日交易数据形成数据集xhyy；从sh.bank中挑选工商银行（601398）的日交易数据形成数据集gsyy。用data步将三个数据集按tdate匹配合并（merge）为数据集portf。计算股票的日收益率，得到变量r1、r2和r3。

(2) 建立(r1,r2,r3)的VAR+DCC模型，用HQC确定VAR模型阶数，DCC各分量模型设定为EGARCH(1，1)。

(3) 将DCC模型中分量EGARCH(1,1)模型的表示杠杆效应的系数的符号约束为负值。

(4) 能够对(3)中的约束条件进行检验吗？

(5) 将DCC模型中r1分量EGARCH模型的ARCH系数和GARCH系数之和约束为0.98。

(6) 对（5）中的约束条件进行检验。

第九章

1．数据集sh_bys_zsyh_r中的变量r_sh、r_b和r_z分别是上证指数、白云山和招商银行的日收益序列。以r_z为观测（应）变量，r_sh为观测方程回归变量，建立时变系数回归CAPM模型。

（1）将状态方程设定为VAR(1)类型，自回归系数矩阵不做约束，误差协方差矩阵对角。

（2）将状态方程设定为VAR(1)类型，自回归系数矩阵对角，误差协方差矩阵不做约束。

（3）将状态方程设定为VAR(1)类型，自回归系数矩阵和误差协方差矩阵均为对角。

（4）将状态方程设定为VAR(1)类型，自回归系数矩阵和误差协方差矩阵均不做约束。

（5）比较（1）-（4）设定的模型的估计结果，你能得出什么结论？

（6）从（2）中模型估计结果"加法离群值汇总"中，确定观测方程异常值的观测序号对应的时间，在测量方程中添加异常值虚拟变量；

（7）在（2）中设定模型state语句中添加选项checkbreak，检测状态变量的结构断点。并在确定断点位置后在状态方程中添加断点虚拟变量。

（8）在state语句中用选项type=rw且不带选项cov()将状态变量设定为常数，估计模型。用估计结果中的"似然计算汇总"给出的轮廓对数似然值和（2）的估计结果的轮廓对数似然值，构造似然比检验，对"回归系数为常数"的原假设进行检验。

（9）用带out=选项的output语句将（2）中模型估计结果保 存为SAS数据集，用proc sgplot画出时变斜率系数的平滑估计和滤波估计的序列图。

2．时间序列变量回归模型可以表示为状态空间模型。设变量y_t, x_t的回归模型为

$$y_t = \beta_0 + \beta_1 x_t + \epsilon_t, \epsilon_t \sim i.i.N(0, \sigma^2)$$

将β_1设为常数状态变量

$$s_t = s_{t-1} + \eta_t, \eta_t \sim N(0,0)$$

回归的状态空间模型具有如下形式

$$y_t = \beta_0 + s_t x_t + \epsilon_t$$
$$s_t = s_{t-1} + \eta_t$$

（1）以数据集sh_trt_r中变量r_trt为因变量，r_m为自变量构造回归模型，将其转换为状态空间模型进行估计。

（2）用proc autoreg模型估计模型参数，与状态空间模型估计结果进行比较。

3．将习题2中模型误差项的独立同分布假设放松为一阶自回归模型，即

$$y_t = \beta_0 + \beta_1 x_t + \epsilon_t$$
$$\epsilon_t = \phi \epsilon_{t-1} + v_t, v_t \sim (0, \sigma^2)$$

定义两个状态变量，一个是常数状态变量s，另一个是误差项状态变量e，把回归模型写成状态空间模型的形式

$$y_t = c + s_t x_t + e_t$$
$$s_t = s_{t-1} + \eta_t, \eta_t \sim N(0,0)$$
$$e_t = \phi e_{t-1} + v_t, v_t \sim N(0, \sigma^2)$$

（1）以数据集sh_trt_r中变量r_trt为因变量，r_m为自变量构造回归模型，模型误差项服从一阶自回归。将其转换为状态空间模型进行估计。

（2）用proc autoreg估计模型，与状态空间模型估计结果进行比较。

3．用数据集 cigar 中变量建立变面板数据模型

$$lsales_{i,t} = \mu + \beta_1 lprice_{i,t} + \beta_2 lndi_{i,t} + \beta_3 lpimin_{i,t} + \gamma_i + \eta_t + \epsilon_{i,t}$$

lprice、lndi 和 lpimin 分别表示对数销售量、对数价格、对数可支配收入和相邻区域的最低对数销售价格，γ_i和η_t分别为个体（区域）效应和时间（年份）效应，ϵ_{it}为模型误差项，满足独立同分布假设。用 proc ssm 设定和估计个体（区域）和时间（年份）双向固定效应模型。

4．为了更好捕捉收益率短端的特征，在 N-S 三因子模型基础上增加一个斜率因子，得出四因子模型动态 N-S 模型

$$y_t(\tau) = L_T + \frac{1-e^{-\lambda_1 \tau}}{\lambda_1 \tau} S_{1,t} + \frac{1-e^{-\lambda_2 \tau}}{\lambda_2 \tau} S_{2,t} + \left[\frac{1-e^{-\lambda_1 \tau}}{\lambda_1 \tau} - e^{-\lambda_1 \tau}\right] C_t + \epsilon_t(\tau)$$

（1）写出四因子动态N-S模型的状态空间模型表示；

（2）用数据yield22的数据估计四因子动态N-S模型，其中参数λ_1和λ_2的值给定，$\lambda_1 = 0.2$、$\lambda_2 = 2.79$。

第十章

1．阅读程序段

```
proc iml;
a=2;b={a,2};
c={1 2,3 4};
d={0.5 0.3};
e={0.1 0.2};
f=a*b;g=a*c;
h=c#d;j=d#c;
k=c#d`;m=c*d`;
```

（1）定义矩阵b的赋值语句是否正确？为什么？

（2）定义矩阵f的语句是否有错误？为什么？

（3）定义矩阵g时，1×1矩阵a和2×2矩阵c能够相乘吗？

（4）矩阵h和j相等吗？为什么？

（5）矩阵k和m相等吗？为什么？

（6）矩阵k和h相等吗？为什么？

2．用索引算符“：”生成矩阵。阅读程序段，回答问题后运行。

```
proc iml;
a=1.2:3.2;
b='A':'F';
c='沈1':'沈10';
d='A1':'F3';
e='沈':'周';
print a,b,c,d;
```

（1）矩阵a是行向量还是列向量？

（2）矩阵d定义正确吗？为什么？

（3）为什么矩阵b定义正确而矩阵e定义不正确？

（4）为什么矩阵c定义正确而矩阵e定义不正确？

3．要用区间分割计算积分$\int_{-\infty}^{\infty} e^{-x^2/2}dx$，用do()算子对积分区间进行划分。

4．1题中，要用矩阵a取值作为矩阵b的第一个元素定义矩阵b，如何实现？

5．设矩阵

$$A=\begin{pmatrix}1 & 3.2 & 5\\ 0 & 0.4 & -0.8\end{pmatrix}$$

（1）有哪些方法可以将矩阵A中绝对值小于1的元素替换为0？写出程序代码。

（2）的矩阵下标降维算符中，<:>、>:<与<>、><的区别是什么？将矩阵A的最小值替换为2的语句？写出将A第二列最大元素替换为4的语句。

（3）用下标降维算符##和矩阵逐元素次方算符##和求和函数sum计算A所有元素平方和。两种计算方法得出的结果相同吗？

6．研究去添加一个观测值对回归系数的影响。以程序10-10为基础，先用x、y的前8个观测求出回归系数OLS估计，在利用逆矩阵更新函数invupdt()，研究添加第 9 个观测后回归系数OLS估计结果的变化。

7．用元素列示法生成矩阵

$$A=\begin{pmatrix}1 & 3.2 & 5\\ 0 & 0.4 & -0.8\\ 1 & 5 & 0.7\end{pmatrix}$$

（1）计算A的行列式值、特征值和迹，验证特征值乘积等于行列式值，特征值和等于迹。

（2）求出A的逆矩阵；
（3）生成与矩阵A相同阶数、元素全为0.5的矩阵。

8．数据集SAShelp.baseball是1986年美国全国棒球运动员数据。
（1）将数据集中变量name、team、nhits和nhome导入为IML变量，要求队员名字以“B”开头、nhits大于100并且nhome大于15；
（2）能将name、team、nhits和nhome导入为IML形成一个矩阵吗？能将name、team导入IML形成一个矩阵吗？如果能写出语句。
（3）use语句和edit语句的区别是什么？用edit、find、read和replace语句将sashelp.baseball中球员名字以“P”开头的球员的nruns的值加1。

9．用循环程序生成行向量a=(1.1 2.2 3.3 … 100.1 101.101)，将其创建为SAS数据集。

10．IML中用sort语句对SAS数据集排序，需要先将数据集读入IML吗？如果要在IML中使用排序数据集标识by组的变量first.vn和last.vn，应如何操作？

11．IML中的选择语句与与DATA步中选择语句有什么不同？

12．IML中得 Do-End循环语句与DATA中的Do-End循环语句有什么不同？

13．什么是符号表？符号表中的符号和对应的矩阵是同一个东西吗？为什么要建立符号表？

14．阅读程序

```
proc iml;
a=10;b=30;c=40;
start f(x) global(b);
  c=x*b;d=x*a;
  return(c);
finish f;
e=f(a);
```

（1）运行时为什么提示矩阵a没有定义（a 0 row 0 col (type ?, size 0)）？
（2）修改程序错误并运行。程序运行后矩阵a、b和c的值是多少？请解释结果。

15．将程序10-23计算积分语句写成函数，并调用函数计算积分$\int_0^1 e^{-x^2/2}dx$。

16．将数据集bteam和ideal读入IML，用IML程序完成程序4-62的查询，将结果输出为SAS数据集。

17．用数据集sh_trt_r中变量r_trt和r_m，建立具有GARCH（1,1）结构的r_trt对r_m的回归模型，误差项服从t分布。（参考例子1和程序10-32）。
（1）将对数似然函数定义为函数
（2）定义参数约束矩阵；
（3）定义opt向量，设定非线性优化是求极大值点；
（4）调用非线性优化子程序nlpnra和nlpdd分别优化对数似然函数，比较结果是否相同。
（5）调用子程序nlpfdd计算参数估计的标准误，并计算t-检验统计量值。

18．R语言函数adfTest()用于一元时间序列的单位根检验，函数arima()用于一元时间序列ARMA模型建模。
（1）将数据集sashelp.stocks中IBM的股票数据转换为R数据框；
（2）对收盘价close及其一阶差分进行单位根检验；
（3）对close的一阶差分序列建立ARMA(4,2)模型。
（4）将模型参数估计结果和程序7-4的结果进行比较。

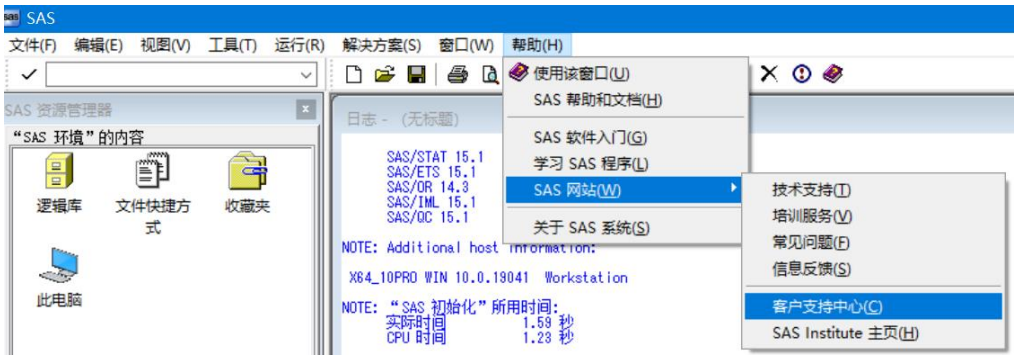
19．用R函数hist做频率直方图
（1）将数据集sashelp.class转换为R数据框class；
（2）将class的第3、4和5列提取为R变量age、weight和height；

（3）用par(mfclo=(2,2))设定一个页面放置2张图，按2行2列摆放；
（4）用hist函数画出age、weight和height的频率直方图。

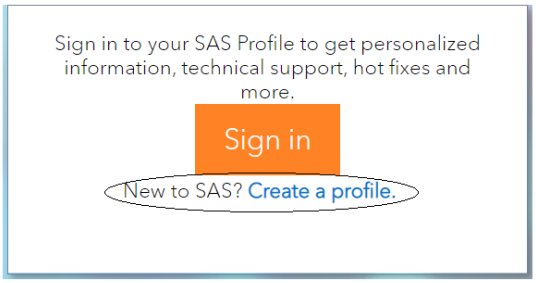
第二部分：参考答案

第一章

1.按如下菜单进入 Sas 主页



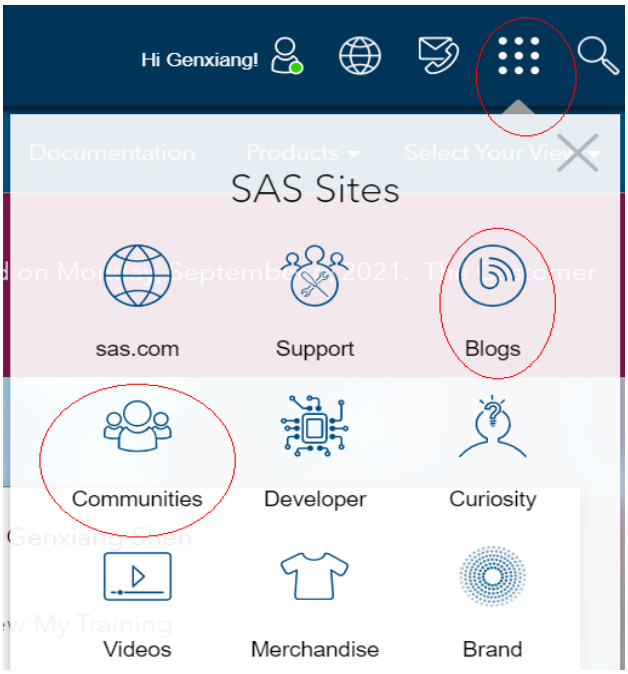
→



按提示注册新用户

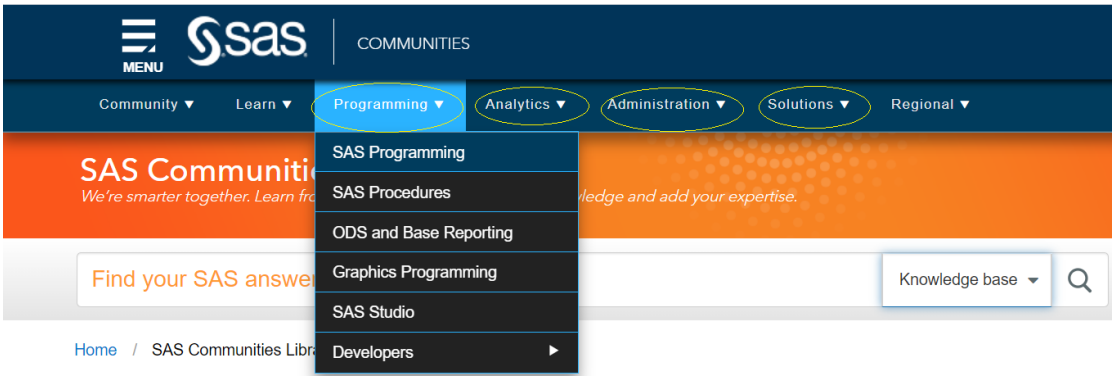
注册完成后进入 SAS 客户支持中心（SAS Customer Support），点击 Communications（Blogs）进入 SAS 社区（博客）

→



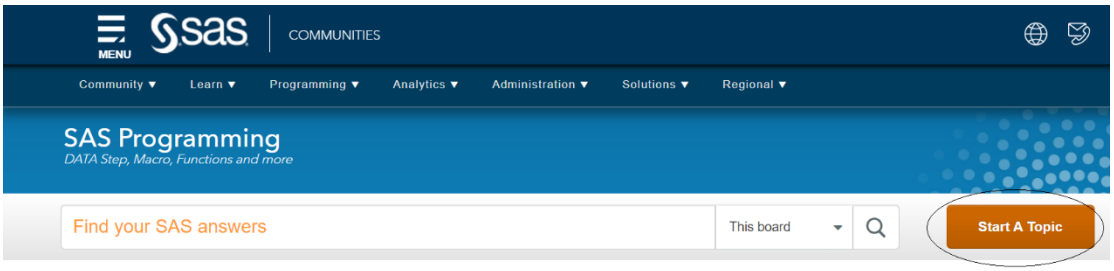
在社区界面选择要进入的社区

→



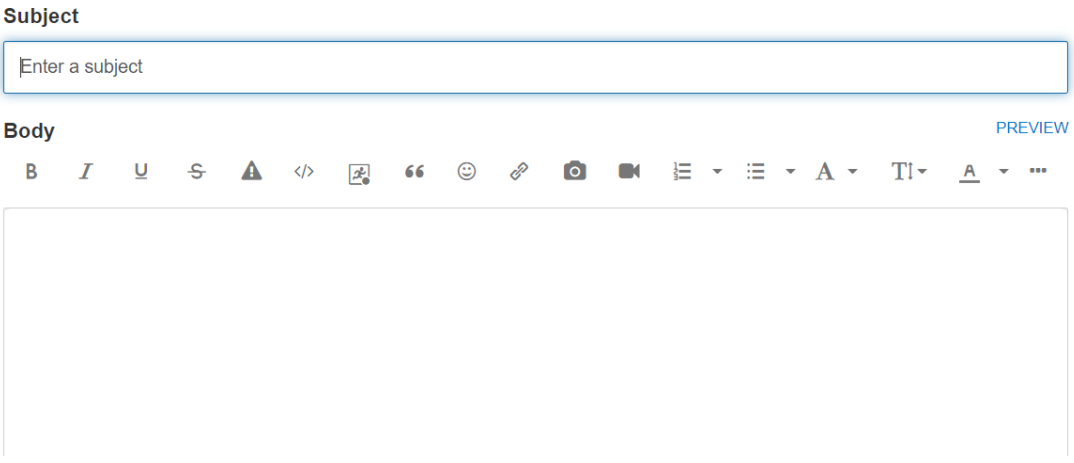
进入 SAS Programming 社区。在 Find your SAS answers 窗口输入你感兴趣的关键词，可以查询有关内容。点击 Start A Topic 可以开始问题讨论

→



点击 Start A Topic 可以开始问题讨论

→

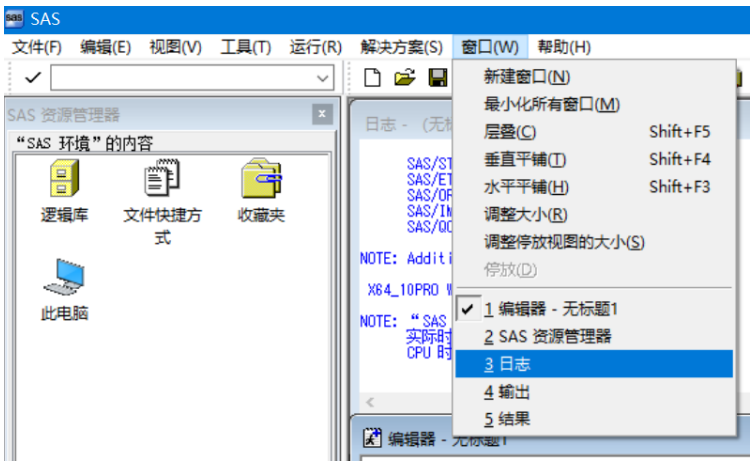


在 Enter a subject 窗口输入问题主题，在 Body 窗口详细描述问题。在页面底部

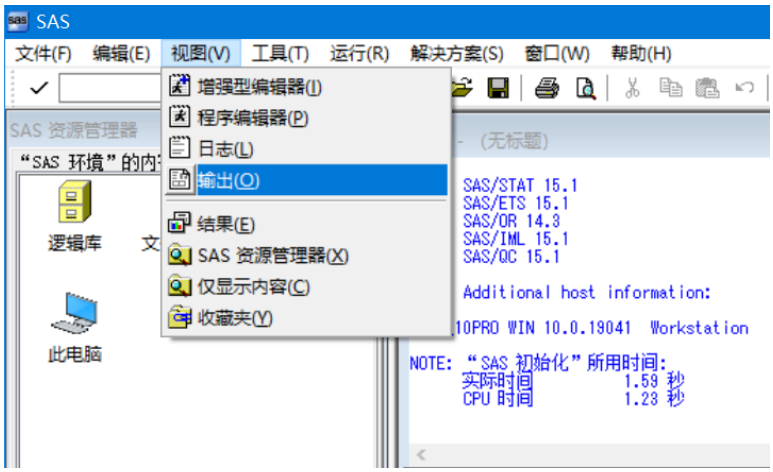
→

点击按钮 Post 可提交问题。

2.SAS 主界面共有五个窗口：日志窗口（Log）、编辑器窗口（Enhenced Editor）、输出窗口（Output）、结果窗口（Result）和资源浏览窗口（Explore）。当前窗口是指对当前操作做出反应的窗口，当前操作的作用窗口。鼠标点击不同窗口可实现当前窗口切换，也可在菜单栏“窗口”下进行切换



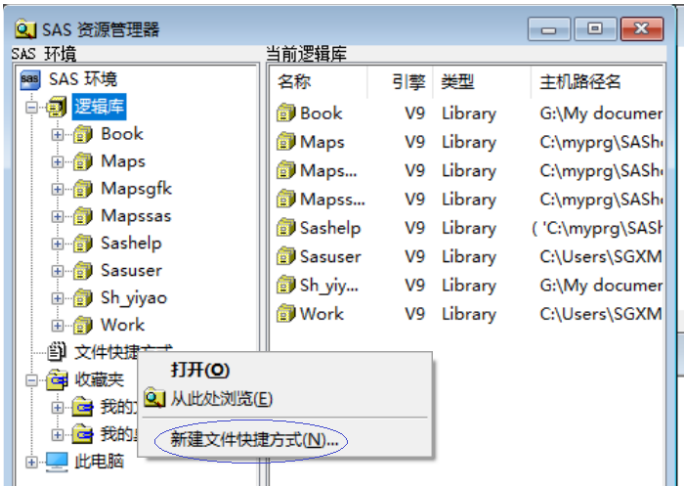
当某个 SAS 窗口不见时，可在菜单栏“窗口”下鼠标点击对应窗口弹出，也可在菜单栏“视图”下鼠标点击对应窗口弹出



3．文件快捷方式建立后，可以在 SAS 主界面对 Windows 下文件进行操作，而不必要切换到 Windows 窗口下。

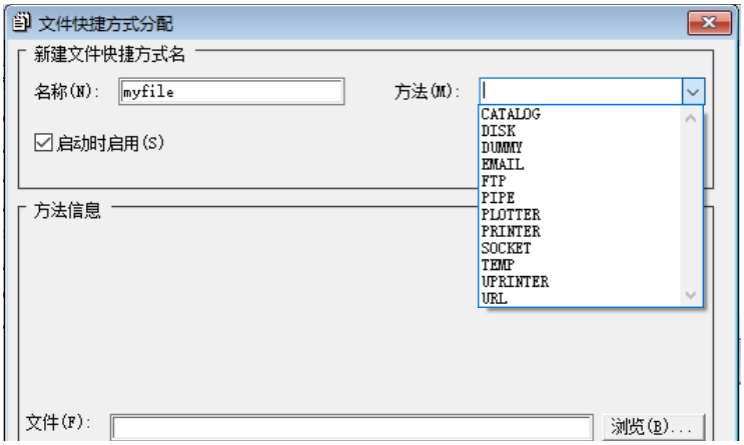
以桌面文件文件 myfile.txt 为例：在资源管理器窗口鼠标 SAS 环境下，鼠标右键“文件快捷方式”图标，在弹出菜单点击“新建文件快捷方式（N）…”

→

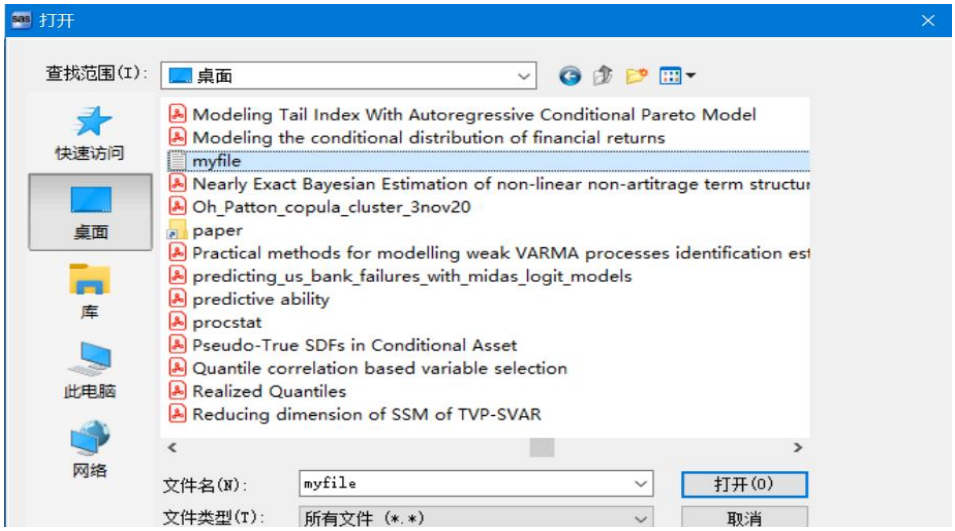


弹出对话框

→



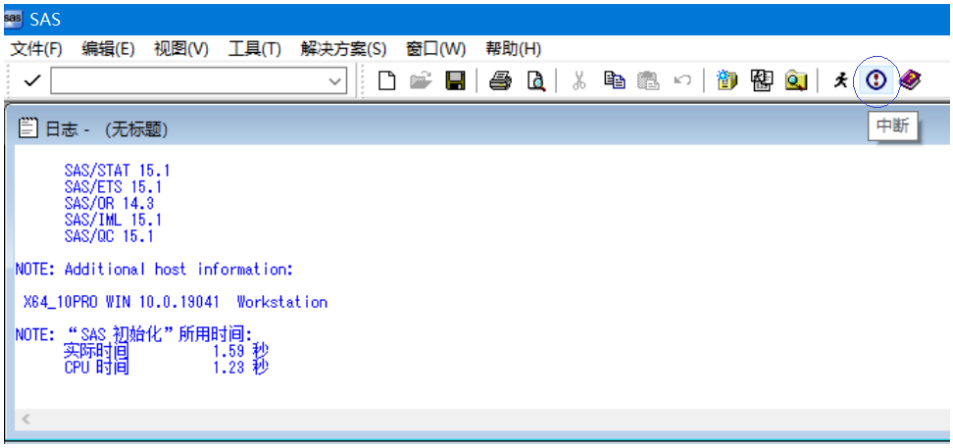
在“名称”窗口内填入快捷文件名，在“方法”后面的备选框中选择“DISK”，并勾选“启动时启用（S）”，然后点击“文件”窗口后的“浏览”按钮，找到文件 myfile.txt



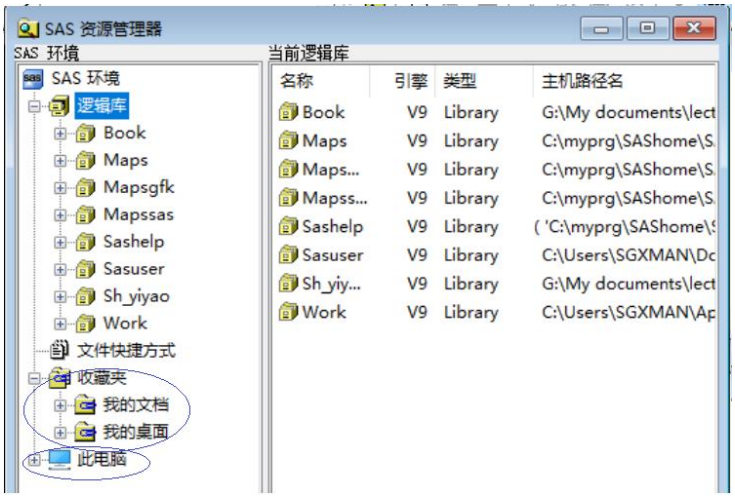
点击“打开”，完成快捷方式建立。注意：默认情况下浏览界面的文件类型为“SAS 程序文件（.sas）”，如要寻找其它类型文件，需要改为“所有文件(*.*)”。

4．工具栏功能中的程序中断（break）功能时菜单栏没有的。

→

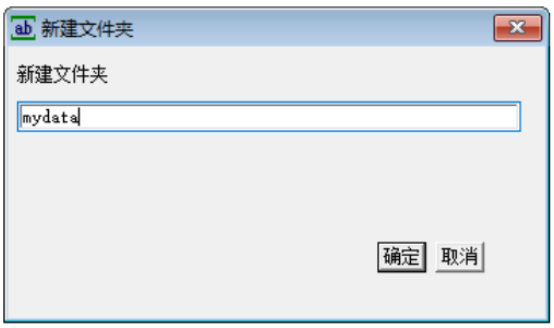


5．SAS 主菜单中的资源管理器窗口提供了“此电脑”图标，使用户再 SAS 界面即可操作 Windows 下的文件。同时在“收藏夹”图标下提供了“我的文档”和“我的桌面”图标，使得 Windows 下的“我的文档”和“桌面”两个文件夹内容的操作更为方便。



操作步骤：点击“此电脑”，右边窗口显示 Windows 硬盘盘符，点击“本地磁盘（D:）”后再弹出的窗口内右键，并选择“新建文件夹(N)”，再弹出的窗口中输入文件夹名“确定”即可。

→



第二章

1．程序代码为

```
proc contents data=sasuser.airports;
run;
```

描述信息输出结果分为三部分

第一部分为数据集的元数据信息：

→

SAS 系统

CONTENTS PROCEDURE

数据集名	SASUSER.AIRPORTS	观测	50
成员类型	DATA	变量	4
引擎	V9	索引	0
创建时间	2021-08-23 16:36:39	观测长度	143
上次修改时间	2021-08-23 16:36:39	删除的观测	0
保护		已压缩	NO
数据集类型		已排序	NO
标签			
数据表示法	WINDOWS_64		
编码	euc-cn Simplified Chinese (EUC)		

第二部分为引擎/主机相关信息

→

引擎/主机相关信息

数据集页面大小	65536
数据集页数	1
首数据页	1
每页最大观测数	457
首数据页的观测数	50
数据集修复数	0
ExtendObsCounter	YES
文件名	C:\Users\SGXMAN\Documents\My SAS Files\9.4\airports.sas7bdat
创建版本	9.0401M6
创建主机	X64_10PRO
所有者名	DESKTOP-VOGE2H0\SGXMAN
文件大小	128KB
文件大小（字节）	131072

从中看出所有者为 DESKTOP-VOGE2H0\SGXMAN，文件路径为 C:\Users\SGXMAN\Document\My SAS Files\9.4\airports.sas7bdat

第三部分为数据集变量属性

→

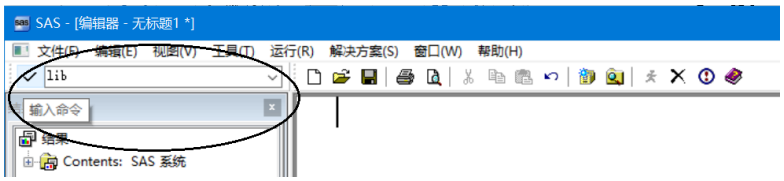
按字母排序的变量和属性列表

#	变量	类型	长度	标签
2	City	字符	50	City Where Airport is Located
1	Code	字符	3	Airport Code
3	Country	字符	40	Country Where Airport is Located
4	Name	字符	50	Airport Name

从中看出，city 的标签为“City Where Airport is Located”。

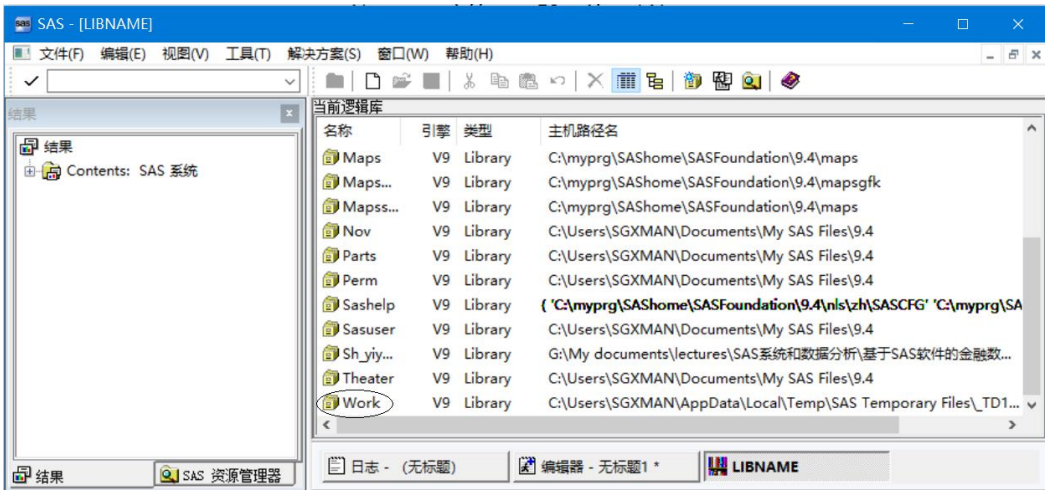
2．在 SAS 命令窗口输入“lib”回车，即

→



弹出逻辑库信息，work 信息位于列表最后一个

→



“主机路径名”下列出的就是各个逻辑库的物理目录。

C:\Users\SGXMAN\AppData\Local\Temp\SAS Temporary Files_TD11008_DESKTOP-VOGE2H0_

采用同样的方法查看重启 SAS 后的 work 对应的物理目录

C:\Users\SGXMAN\AppData\Local\Temp\SAS Temporary Files_TD5980_DESKTOP-VOGE2H0_

与上次的物理目录并不一样。因为 Work 目录在 SAS 启动时随机产生临时逻辑库，其对应的物理目录位置固定，但名称是变化的。

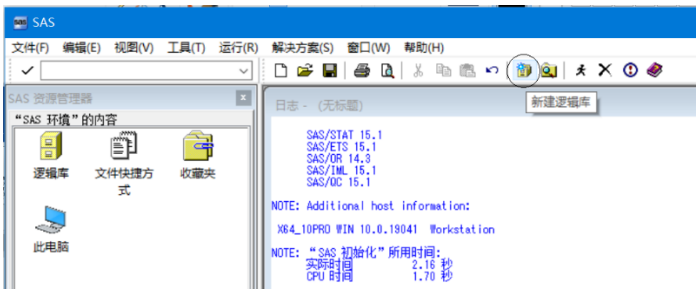
3．建立逻辑库有三种途径：libname 命令、快捷键和菜单操作。

libname 命令：在编辑窗口提交命令

libname sasdata “D:\lecture\SAS\mydata”

快捷键途径：在工具栏点击图标

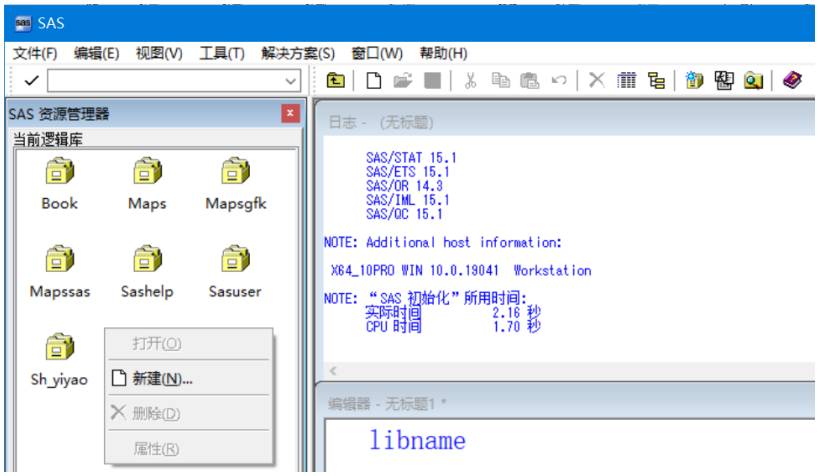
→



按指引建立逻辑库

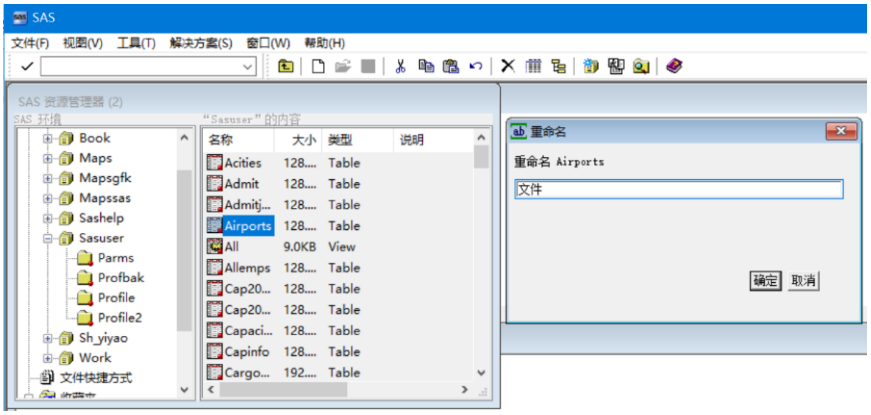
菜单操作：在“SAS 环境”窗口点击“逻辑库”图标进入“当前逻辑库”界面，在界面点击鼠标右键，

→



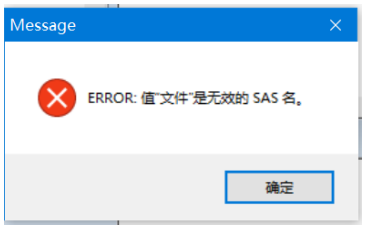
在弹出的菜单中点击“新建(N)”，弹出“新建逻辑库”界面，按导引建立逻辑库。

4．以文件 sasuser.airports 为例：在 sasuer 逻辑库下，鼠标右键点击文件 airports，在弹出的菜单中选择“重命名”，弹出对话框窗口中输入“文件”点击“确定”。



弹出提示信息

→



因为汉字不能作为文件名。不能给数据集设置标签。

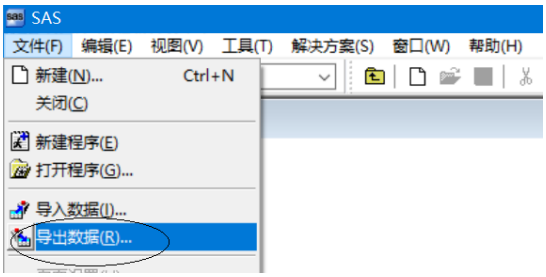
5．“万能”日期输入格式 ANYDTDTE.将外部数据 02/03/12 读为 2002 年 3 月 12 日。用输入格式 mmddyy10.可读为 2012 年 2 月 3 日，用输入格式 ddmmyy10.可读为 2012 年 3 月 2 日.可用如下 data 步程序进行验证

```
data test;
input date1 anydtdte. @;
input @1 date2 mmddyy10.@;
input @1 date3 ddmmyy10.;
format date1 date2 date3 date8.;
datalines;
02/03/12
  ;
```

6．数据格式 COMMA8.2 可将外部数据($7.8)读作-7.8。

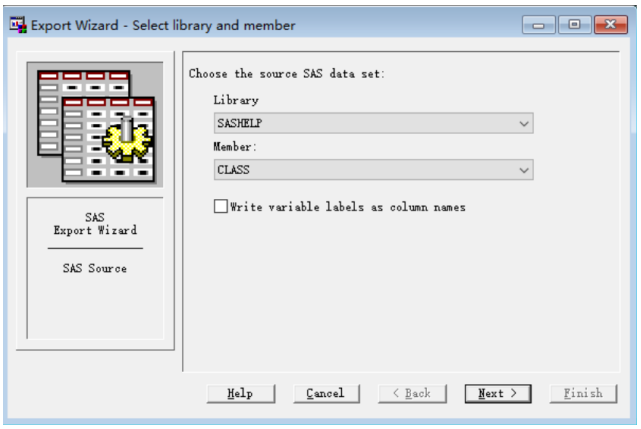
7．采用 SAS 主界面菜单“文件”下的“导出数据”

→



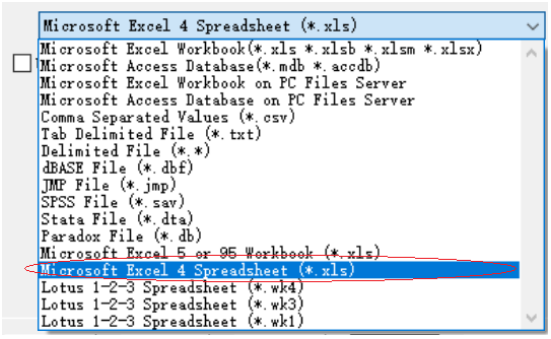
在弹出的对话框中“Library”选择 SASHELP，“Number”选择 CLASS

→



点击“Next”后，在弹出的对话框中“What type of data do you wish to export”下方列表中选择“Microsoft Excel4 Spreadsheet（*.xls）”

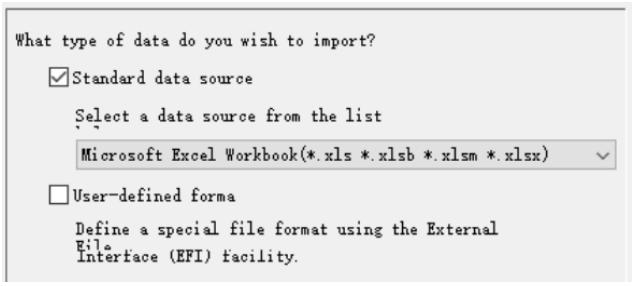
→



选择导出文件的保存目的地，确定后完成导出。

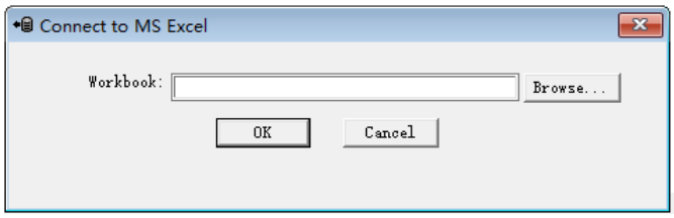
“文件”下拉菜单中“导入数据”，在“What type of data do you wish to import”下的“Standard data source”下的文件类型列表选择“Microsoft Excel Workbook”

→



弹出“Connect to MS Excel”窗口

→



通过“Browse”找到要导入的文件 class.xls。选择导入的逻辑库（work）和 SAS 文件名（class）后点击“Finish”，在日志窗口出现如下提示

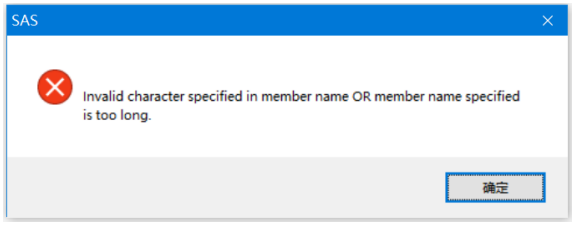
ERROR：文件“_IMEX_.’class$’n.DATA”不存在。

ERROR：导入失败。详细信息，请参阅“SAS 日志”。

```
NOTE: 成功创建“C:\Users\SGXMAN\Desktop\class.xls”文件。
ERROR: 文件“_IMEX_.'class$'n.DATA”不存在。
ERROR: 导入失败。详细信息，请参阅“SAS 日志”。
```

在 Work 逻辑库没有出现导入的数据集。解决的办法是首先用 Excel 打开文件 class.xls，将其“另存为”CSV 文件 class.csv，然后再将其导入 SAS 即可。

给出 SAS 数据集名字“123”时，会弹出如下信息



提示输入的文件名不合法。因为文件名不能以数字开头。

第三章

1．表达式 3+’2’不正确，但可行。不正确的原因是数值 3 不能和字符’2’相加，可行的原因是 SAS 会将字符’2’自动转换为数值，然后和 3 相加。表达式 3+’a’既不正确也不可行。不正确是因为数值 3 不能和字符’a’相加，不可行是因为 SAS 不会将字符’a’自动转换为数值。’2018/05/03’d 不是正确的日期常量表达式，因为尽管日期常量用日期字符加上 d 的方法表达，但日期字符必须是特殊的日期形式：DDMMMYY。’3Feb19’d 是正确的日期常量表达，SAS 将其中的’19’识别为年份。

2．SAS 在编译阶段对语句 y=sim(x);进行编译时，无法找到函数 sim(x)，信息窗口给出提示信息：

ERROR 68-185: 函数 SIM 未知，或无法访问。

终止执行并在信息窗口给出错误信息

WARNING: 数据集 ××× 可能不完整。该步停止时，共有 0 个观测和 3 个变量。

SAS 在编译时将 sim(x)识别为函数而不是变量，因为变量名称不允许出现字符”()”。

3．a)函数 logistic()的功能是对自变量实施逻辑变换，即

$$\text{logistic}(x)=\frac{e^x}{1+e^x}$$

例如 data 步

```
data one;
   x=0.5;
   z=logistic(x);
   put z=;
run;
```

用函数 logistic()对变量 x 实施变换，定义变量 z。运行结果为

z=0.6224593312

b)函数 fact()的功能是计算自变量的阶乘（facorial），即

$$\text{fact}(x)=x!\ ,x>0$$

例如 data 步

```
data one;
   x=fact(5);
   put x=;
run;
```

计算 5 的阶乘赋值给变量 x，put 语句的输出结果为

x=120

c)substr()的功能是截取自变量字符串的子串，substr()=的功能是用给定字符串替换自变量字符串中的某一部分，使用格式为

substr(x,n1<,n2>)=str

功能：将 x 中从左边第 n1 个字符开始的 n2 个字符替换为字符 str，str 可以是字符常量，也可以是字符变量。n1 和 n2 可以是数值常数、数值变量或者表达式，n2 为可选项，缺省替换范围从第 n1 个字符到字符结束。

例如 data 步

```
data new;
```

```
a='KIDNAP';
substr(a, 1, 3)='CAT';
put a=;
b=a;
substr(b, 4)='TY';
put b=;
run;
```

将变量 a 中从第一个字符开始的 3 个字符替换为'CAT'，把变量 b 从第四个字符开始的所有字符替换为'TY'。运行结果为

```
a=CATNAP
b=CATTY
```

d) Yrdif()的功能是计算两个特定日期间隔的年数，也可以计算年龄。函数的格式为

```
Yrdif(date1,date2<,basis>)
```

date1 和 date2 分别为起止日期，basis 为可选项，确定计算方法。

例如程序

```
data _null_;
 sdate='16oct1998'd;
 edate='16feb2010'd;
 y30360=yrdif(sdate, edate, '30/360');
 yactact=yrdif(sdate, edate, 'ACT/ACT');
 yact360=yrdif(sdate, edate, 'ACT/360');
 yact365=yrdif(sdate, edate, 'ACT/365');
 put y30360= / yactact= / yact360= / yact365= ;
run;
```

计算 1988 年 10 月 16 日到 2010 年 2 月 16 日之间间隔的年时，采用四种不同计算方法，输出结果如下：

```
y30360=11.333333333
yactact=11.336986301
yact360=11.502777778
yact365=11.345205479
```

程序

```
data _null_;
   sdate='16oct1998'd;
   edate='16feb2010'd;
   age=yrdif(sdate, edate, 'AGE');
   put age= 'years';
run;
```

计算 1988 年 10 月 16 日出生的人到 2010 年 2 月 16 日的年龄，输出结果为

```
age=11.336986301 years
```

4. if name='罗';不行，因为此时比较的时名字的全部是否为'罗'，正确的写法是

```
if name=:'罗';
```

也可采用函数 substr()将语句写为

```
if substr(name,1,2)='罗';
```

因为一个汉字的字符长度为 2，因此函数的第二个自变量为 2。

采用 where 语句也可以用两种方式：

```
where name=:'罗';
```

和

```
where substr(name,1,2)='罗';
```

语句 where name not like '%德';选出的是姓名中最后一个汉字不是'德'的人。

5．空语句是指只有语句结束标志';'的语句。当 data 步包含内嵌数据时，作为内嵌数据行结束标志的空语句，可以作为 data 步的结束边界。

6．自动变量_Error_和_N_的初次赋值是在执行阶段。运行一下 data 步程序

```
data test3;
put _all_;
x=1;
y=1+'a';
/*sat sashelp.class;*/
put _all_;
run;
```

信息窗口输出结果为

```
x=. y=. _ERROR_=0 _N_=1
NOTE: Invalid numeric data, 'a' , at 行 86 列 5.
x=1 y=. _ERROR_=1 _N_=1
x=1 y=. _ERROR_=1 _N_=1
```

程序没有语法错误，data步运行1次。第一个put语句输出的自动变量值分别为_ERROR_=0 _N_=1，x和y的值均为缺失，第二个put语句输出的自动变量值分别为_ERROR_=1 _N_=1，x和y的值分别为1和缺失值。

运行程序

```
data test5;
put _all_;
x=1;
y=1+'a';
E=_Error_;
N=_N_;
sat sashelp.class;
put _all_;
run;
```

由于发生语法致命错误，DATA 步没有执行，生成的数据集 test5 中变量 x、y、E 和 N 均为空，因为此时赋值语句 E=_Error_没有执行，不能判断_Error_是否已赋初始值。

7．声明（declarative）语句在编译环节起作用，向 SAS 提供必要的信息。DATA 步开始边界的 DATA 语句、设置输入数据集观测过滤条件 where 语句、标明其下面为数据行的 datalines 语句以及改变变量输入输出格式的 informat/format 语句等，是声明语句。执行（executable）语句在执行环节起作用，在 DATA 步每次内置循环中产生一些动作。读入外部数据的 input 语句、条件成立执行后续语句的 if 语句、打开 SAS 集读取观测的 set 语句以及停止当前 DATA 步执行的 stop 语句等，是执行语句。

8．data 步程序

```
data test;
set sashelp.class(keep=name sex);
```

```
h=height*0.025;
run;
```

中的错误在于数据集选项 keep=语句的位置。由于编译阶段执行了输入数据集选项，只有变量 name 和 sex 在 PDV 中运行阶段将数据观测读入，尽管编译赋值语句 h=height*0.025;时在 PDV 中安置了变量 height，但执行阶段没有把数据集中数据读给 height，因此赋值语句 h=height*0.025;执行时 height 没有初始化。程序执行后的信息窗口提示

NOTE: 变量 height 未初始化。

NOTE: 缺失值的生成是对缺失值执行操作的结果。

修改的方法是将keep=选项放在输出数据集后面，即

```
data test(keep=name sex);
set sashelp.class;
h=height*0.025;
run;
```

程序

```
data test;
set sashelp.class;
h=height*0.025;
where h<10;
run;
```

的错误在于where语句中的变量必须是数据集变量，因为where语句为声明语句，在编译阶段起作用，但编译阶段变量h未初始化。执行时信息窗口的提示信息为

ERROR: 变量 h 不在文件“SASHELP.CLASS”中。

修改方法是将where语句改为if语句，即

```
data test;
set sashelp.class;
h=height*0.025;
if h<10;
run;
```

9. 不存在语法错误，程序能够运行，但数据集 s 没有观测。原因在于 if 语句执行时没有打开数据集读取观测，此时变量 sex 值为缺失值，if 语句的条件不满足，后续语句不被执行。修改方法是将 if 语句放在 set 语句之后，即

```
data s;
set sashelp.class;
if sex='男';
h=height*0.025;
run;
```

10. 字符常量可以用 input()函数转换为日期常量，只要在输入格式自变量位置给出日期格式即可，即

```
input('03apr2019',date10.)
```

put()函数可以把数值和字符转换为数值。日期常量是数值常量的一种格式，函数 put()可以把日期转换为日期，只需要在输出格式自变量处给出日期格式即可，即

```
input('03apr2019'd,date10.)
```

相当于将日期从一种格式转为另一种格式。

要将日期常量转换为字符常量，用 put()函数且输出格式自变量的值与源数据格式相

同，即

```
input('03apr2019'd,date9.)
```

11.先生成数据集test

```
data test;
input city $ year;
datalines;
上海 2017
上海 2018
上海 2019
;
```

采用data步生成数据集test1

```
data test1;
if _n_=1 then
do;
set test;
year1=year;
p=2;
set test point=p;
year2=year;
set test point=p;
year3=year;
end;
output;
drop year;
stop;
run;
```

主要是set语句选项point=的使用，使得set语句能够按需要随机读取观测。

12.运行程序后，信息窗口出现如下信息

```
Name=  Sex=  Age=. Height=. Weight=. _ERROR_=0 _N_=1
Name=阿尔弗雷德 Sex=男 Age=14 Height=69 Weight=112.5 _ERROR_=0 _N_=1
Name=阿尔弗雷德 Sex=男 Age=14 Height=69 Weight=112.5 _ERROR_=0 _N_=2
Name=爱丽丝 Sex=女 Age=13 Height=56.5 Weight=84 _ERROR_=0 _N_=2
Name=爱丽丝 Sex=女 Age=13 Height=56.5 Weight=84 _ERROR_=0 _N_=3
Name=芭芭拉 Sex=女 Age=13 Height=65.3 Weight=98 _ERROR_=0 _N_=3
Name=芭芭拉 Sex=女 Age=13 Height=65.3 Weight=98 _ERROR_=0 _N_=4
Name=凯露 Sex=女 Age=14 Height=62.8 Weight=102.5 _ERROR_=0 _N_=4
Name=凯露 Sex=女 Age=14 Height=62.8 Weight=102.5 _ERROR_=0 _N_=5
Name=亨利 Sex=男 Age=14 Height=63.5 Weight=102.5 _ERROR_=0 _N_=5
Name=亨利 Sex=男 Age=14 Height=63.5 Weight=102.5 _ERROR_=0 _N_=6
Name=詹姆斯 Sex=男 Age=12 Height=57.3 Weight=83 _ERROR_=0 _N_=6
Name=詹姆斯 Sex=男 Age=12 Height=57.3 Weight=83 _ERROR_=0 _N_=7
Name=简 Sex=女 Age=12 Height=59.8 Weight=84.5 _ERROR_=0 _N_=7
Name=简 Sex=女 Age=12 Height=59.8 Weight=84.5 _ERROR_=0 _N_=8
Name=雅妮特 Sex=女 Age=15 Height=62.5 Weight=112.5 _ERROR_=0 _N_=8
```

```
Name=雅妮特 Sex=女 Age=15 Height=62.5 Weight=112.5 _ERROR_=0 _N_=9
Name=杰弗瑞 Sex=男 Age=13 Height=62.5 Weight=84 _ERROR_=0 _N_=9
Name=杰弗瑞 Sex=男 Age=13 Height=62.5 Weight=84 _ERROR_=0 _N_=10
Name=约翰 Sex=男 Age=12 Height=59 Weight=99.5 _ERROR_=0 _N_=10
Name=约翰 Sex=男 Age=12 Height=59 Weight=99.5 _ERROR_=0 _N_=11
Name=乔伊斯 Sex=女 Age=11 Height=51.3 Weight=50.5 _ERROR_=0 _N_=11
Name=乔伊斯 Sex=女 Age=11 Height=51.3 Weight=50.5 _ERROR_=0 _N_=12
Name=茱迪 Sex=女 Age=14 Height=64.3 Weight=90 _ERROR_=0 _N_=12
Name=茱迪 Sex=女 Age=14 Height=64.3 Weight=90 _ERROR_=0 _N_=13
Name=罗伊斯 Sex=女 Age=12 Height=56.3 Weight=77 _ERROR_=0 _N_=13
Name=罗伊斯 Sex=女 Age=12 Height=56.3 Weight=77 _ERROR_=0 _N_=14
Name=玛丽 Sex=女 Age=15 Height=66.5 Weight=112 _ERROR_=0 _N_=14
Name=玛丽 Sex=女 Age=15 Height=66.5 Weight=112 _ERROR_=0 _N_=15
Name=菲利普 Sex=男 Age=16 Height=72 Weight=150 _ERROR_=0 _N_=15
Name=菲利普 Sex=男 Age=16 Height=72 Weight=150 _ERROR_=0 _N_=16
Name=罗伯特 Sex=男 Age=12 Height=64.8 Weight=128 _ERROR_=0 _N_=16
Name=罗伯特 Sex=男 Age=12 Height=64.8 Weight=128 _ERROR_=0 _N_=17
Name=罗纳德 Sex=男 Age=15 Height=67 Weight=133 _ERROR_=0 _N_=17
Name=罗纳德 Sex=男 Age=15 Height=67 Weight=133 _ERROR_=0 _N_=18
Name=托马斯 Sex=男 Age=11 Height=57.5 Weight=85 _ERROR_=0 _N_=18
Name=托马斯 Sex=男 Age=11 Height=57.5 Weight=85 _ERROR_=0 _N_=19
Name=威廉 Sex=男 Age=15 Height=66.5 Weight=112 _ERROR_=0 _N_=19
Name=威廉 Sex=男 Age=15 Height=66.5 Weight=112 _ERROR_=0 _N_=20
```

（1） 两次内容的不同表现在两个地方：a)第一次内置循环时，第一个 put _all_输出的是 set 语句没有执行时各个变量的当前值，自动标量_N_=1，_Error_=0，数据集变量取缺失值。第二个 put _all_输出的是 set 语句执行后各变量当前值，此时数据集变量已被读入数据集观测值。b)第一次之后的内置循环中，第一个 put _all_输出的是 set 语句没有执行时各个变量的当前值，数据集变量值是上一个循环的保留值（retain），第二个 put _all_输出的是 set 语句执行后各变量当前值，此时数据集变量已被读入数据集的当前观测值。

（2） 这与有 set 语句的 data 内置循环机理有关：当第 19 次循环完成后，data 进入第 20 次循环，此时_N_的值已经是 20，但 set 语句读到文件结束标志，data 结束，第 20 次循环的第二个 put _all_没有执行。据此判断_N_的值实在 data 当前循环结束下一次开始时增加的。

将程序改为

```
data test;
put _all_;
set sashelp.class;
_n_=3;
put _all_;
run;
```

运行后信息窗口的信息为

```
Name=  Sex=  Age=. Height=. Weight=. _ERROR_=0 _N_=1
Name=阿尔弗雷德 Sex=男 Age=14 Height=69 Weight=112.5 _ERROR_=0 _N_=3
```

Name=阿尔弗雷德 Sex=男 Age=14 Height=69 Weight=112.5 _ERROR_=0 _N_=2
Name=爱丽丝 Sex=女 Age=13 Height=56.5 Weight=84 _ERROR_=0 _N_=3
Name=爱丽丝 Sex=女 Age=13 Height=56.5 Weight=84 _ERROR_=0 _N_=3
Name=芭芭拉 Sex=女 Age=13 Height=65.3 Weight=98 _ERROR_=0 _N_=3
Name=芭芭拉 Sex=女 Age=13 Height=65.3 Weight=98 _ERROR_=0 _N_=4
Name=凯露 Sex=女 Age=14 Height=62.8 Weight=102.5 _ERROR_=0 _N_=3
Name=凯露 Sex=女 Age=14 Height=62.8 Weight=102.5 _ERROR_=0 _N_=5
Name=亨利 Sex=男 Age=14 Height=63.5 Weight=102.5 _ERROR_=0 _N_=3
Name=亨利 Sex=男 Age=14 Height=63.5 Weight=102.5 _ERROR_=0 _N_=6
Name=詹姆斯 Sex=男 Age=12 Height=57.3 Weight=83 _ERROR_=0 _N_=3
Name=詹姆斯 Sex=男 Age=12 Height=57.3 Weight=83 _ERROR_=0 _N_=7
Name=简 Sex=女 Age=12 Height=59.8 Weight=84.5 _ERROR_=0 _N_=3
Name=简 Sex=女 Age=12 Height=59.8 Weight=84.5 _ERROR_=0 _N_=8
Name=雅妮特 Sex=女 Age=15 Height=62.5 Weight=112.5 _ERROR_=0 _N_=3
Name=雅妮特 Sex=女 Age=15 Height=62.5 Weight=112.5 _ERROR_=0 _N_=9
Name=杰弗瑞 Sex=男 Age=13 Height=62.5 Weight=84 _ERROR_=0 _N_=3
Name=杰弗瑞 Sex=男 Age=13 Height=62.5 Weight=84 _ERROR_=0 _N_=10
Name=约翰 Sex=男 Age=12 Height=59 Weight=99.5 _ERROR_=0 _N_=3
Name=约翰 Sex=男 Age=12 Height=59 Weight=99.5 _ERROR_=0 _N_=11
Name=乔伊斯 Sex=女 Age=11 Height=51.3 Weight=50.5 _ERROR_=0 _N_=3
Name=乔伊斯 Sex=女 Age=11 Height=51.3 Weight=50.5 _ERROR_=0 _N_=12
Name=茱迪 Sex=女 Age=14 Height=64.3 Weight=90 _ERROR_=0 _N_=3
Name=茱迪 Sex=女 Age=14 Height=64.3 Weight=90 _ERROR_=0 _N_=13
Name=罗伊斯 Sex=女 Age=12 Height=56.3 Weight=77 _ERROR_=0 _N_=3
Name=罗伊斯 Sex=女 Age=12 Height=56.3 Weight=77 _ERROR_=0 _N_=14
Name=玛丽 Sex=女 Age=15 Height=66.5 Weight=112 _ERROR_=0 _N_=3
Name=玛丽 Sex=女 Age=15 Height=66.5 Weight=112 _ERROR_=0 _N_=15
Name=菲利普 Sex=男 Age=16 Height=72 Weight=150 _ERROR_=0 _N_=3
Name=菲利普 Sex=男 Age=16 Height=72 Weight=150 _ERROR_=0 _N_=16
Name=罗伯特 Sex=男 Age=12 Height=64.8 Weight=128 _ERROR_=0 _N_=3
Name=罗伯特 Sex=男 Age=12 Height=64.8 Weight=128 _ERROR_=0 _N_=17
Name=罗纳德 Sex=男 Age=15 Height=67 Weight=133 _ERROR_=0 _N_=3
Name=罗纳德 Sex=男 Age=15 Height=67 Weight=133 _ERROR_=0 _N_=18
Name=托马斯 Sex=男 Age=11 Height=57.5 Weight=85 _ERROR_=0 _N_=3
Name=托马斯 Sex=男 Age=11 Height=57.5 Weight=85 _ERROR_=0 _N_=19
Name=威廉 Sex=男 Age=15 Height=66.5 Weight=112 _ERROR_=0 _N_=3
Name=威廉 Sex=男 Age=15 Height=66.5 Weight=112 _ERROR_=0 _N_=20

由于_N_是可以编程中使用的变量，在 data 步中可以用赋值语句改变取值，但其 data 步运行次数的计数功能并未改变，当前 data 步结束下一次 data 开始时，_N_的值重新自动改为计数次数。

13.提交程序

```
data _null_;
set sashelp.class;
rename height=h;
```

```
keep name age sex;
run;
```

由于用自动变量_null_作为输出数据集名，data 步不生成数据集，即没有输出数据集，而 rename 只对 PDV 输出到输出数据集的变量起作用，因此没有出现 keep 语句和 rename 语句的冲突，程序 3-31 输出数据集为 null，出现了 keep 语句和 rename 语句的冲突。

14. 在第一条记录前添加空白记录，要在第一次循环中 set 语句打开数据集前用 output 语句输出空白记录，即

```
data test;
if _n_=1 then output;
set sashelp.class;
output;
run;
```

output 语句屏蔽了 data 的自动输出功能，第二个 output 语句保证了 class 中观测的正常输出。

在第 5 条观测后添加空白记录的程序为

```
data test2;
set sashelp.class;
if _n_=5 then
do;
output;
Name='';sex='';Age=.;
height=.;weight=.;
end;
output;
run;
```

15. 程序代码为

```
data counting;
set sashelp.class(rename=(age=age1 name=name1)) nobs=num;
n=0;
do i=1 to num;
set sashelp.class point=i;
if age>age1 then n+1;
end;
run;
```

16. 程序代码为

```
data blank;
set sashelp.class;
if _n_=4 then
do i=1;
output;
name='';sex='';
age=.;weight=.;height=.;
end;
```

```
output;
run;
```

注意其中需要两个 output 语句，第一个将第四条记录输出，否则将被空白记录覆盖掉，第二个将当前记录输出，如果没有第二个 output 语句，由于 data 步的自动输出功能失效，则只有第四条记录被输出。

循环指标变量 i 为 data 步定义的变量，在内置循环中不具有 retain 功能，每次内置循环开始将被初始化为缺失值，因此 do-end 循环结束后，从第 5 次内置循环开始，i 的值一直为缺失值。

17. 程序

```
data x;
do i=1 to 25;
set sashelp.class point=i;
output;
end;
stop;
run;
```

（1） 如果没有 stop 语句，则由于 point=i 将 set 语句记录指针指向特定观测，达不到文件结束标志，data 不会结束，进入死循环。

（2） do-end 循环中，i=20 到 i=25 时，point=i 指向的观测超出了数据集观测范围，不能读取数据，而数据集变量的 retain 功能则将第 19 次循环时的变量值在此后的每次循环中保留，输出的均为第 19 条观测。

18. 数据集中没有观测的原因是 data 中的 stop 语句在 data 步到达结束边界语句 run; 之前便结束了运行，没有实现自动输出。如果在 stop 之前添加 output 语句，即

```
data x1;
n=23;
set sashelp.class point=n;
output;
stop;
run;
```

则 x1 中有一条空白观测，每个变量取值为缺失，因为记录指针指向 23 超出了数据集观测范围，不能读取数据，输出的变量仍然是编译时的取值（缺失值）。

19. （1）test 和 sashelp.class 的区别在于数据集变量的排列数序发生了变化，中的变量顺序为 age、weight、height、sex 和 name。

（2）程序中的 do-end 程序块语句没有执行，因为 if 语句的条件不会满足，其作用是在编译阶段 set 语句将 age、weight 和 height 安置在 PDV，而 sex 和 name 则不读入，retain 语句编译将 sex 和 name 安置在 PDV，从而实现变量 PDV 中变量次序的改变；

(3) 第二个 set 语句把观测读入 PDV。

20. 修改后的程序为

```
proc sort;
by stkcd;
run;
data test1;
set test;
by stkcd;
if last.stkcd then
do;
```

```
output;
stkcd='';date=.;clp=.;
end;
output;
run;
```

21. （1）

```
data temp1;
set book.cpi;
if month(date) in (3,6,9,12);
run;
```

（2）先将 rate 中数据按月进行排序，

```
data temp2;
set book.rate;
m=month(date);
run;
proc sort data=temp2;
by y m;
run;
```

然后从中抽取每月最后一条记录

```
data temp3;
set temp2;
by y m;
if last.m;
if month(date) in (3,6,9,12);
run;
```

（3）将 temp1 和 temp3 按 date 进行匹配合并

首先将 temp1 中 1954 年 9 月前的数据删除

```
data temp4;
set temp1;
if date>='01sep1954'd;
run;
```

然后实施合并。第一种合并为非匹配合并

```
data final;
merge temp4 temp3;
keep date cpi rate;
run;
```

第二种合并为匹配合并，按年份和季度进行合并。

```
data temp5;
set temp4;
y=year(date);
m=month(date);
run;
data temp6;
set temp3;
```

```
y=year(date);
m=month(date);
run;
proc sort data=temp5;
by y m;
proc sort data=temp6;
by y m;
data final2;
merge temp5 temp6;
by y m;
drop y m;
run;
```

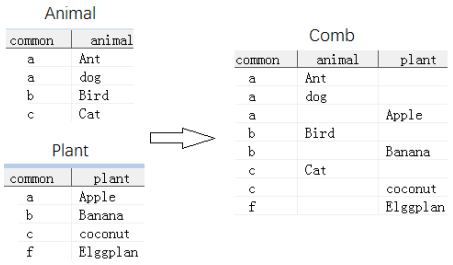
22．将数据集 Animal 和 Plant 非匹配并接的代码为

```
data comb1;
merge book.animal book.plant;
run;
```

合并后的数据集 comb1 为

	common	animal	plant
1	a	Ant	Apple
2	b	dog	Banana
3	c	Bird	coconut
4	f	Cat	Elggplan

程序 3-70 合并后得到的数据集 comb 为



通过比较看出，非匹配 merge 并接是将两个数据集左右“并放”，相当于按观测序号的并接，而用程序 3-70 用 set 语句以变量 common 变量的匹配并接，则按照 common 的值进行匹配。

23．程序代码为

```
data c_Num(drop=i);
no=0;
do i=1 to 7;
no=i;
output;
end;
data class_num;
if 0 then
do;
set sashelp.class nobs=p1;
set c_num nobs=p2;
end;
```

```
do i=1 to p1;
set sashelp.class point=i;
set c_num point=p2;
output;
end;
drop i;
stop;
run;
```

24．将程序 3-75 中的 if 语句位置改编为

```
data class_num2;
set sashelp.class;
set c_num point=_n_;
if name=:'罗';
run;
```

运行得出的数据集 class_num2 为

	Name	Sex	Age	Height	Weight	no
1	罗伊斯	女	12	56.3	77	7
2	罗伯特	男	12	64.8	128	7
3	罗纳德	男	15	67	133	7

程序 3-75 运行得出的数据集 class_num1 为

	Name	Sex	Age	Height	Weight	no
1	罗伊斯	女	12	56.3	77	.
2	罗伯特	男	12	64.8	128	.
3	罗纳德	男	15	67	133	.

区别在于变量 no 的值。if 语句的位置没有移动之前，只有到第 7 次循环 if 语句条件才满足，其后的 set 语句执行，但此时 _n_=7，point=_n_给出的记录指针值超出数据集 c_num 的记录范围，set 语句不能读取数据，no 变量的值为缺失值（编译阶段的值）。if 语句移动后，第二个 set 语句被执行，no 的值成功读取，只有到第 7 次循环 if 语句条件满足，data 步采用继续运行到 run;语句并执行自动输出功能，此时的 no 值为 c_num 的第 7 条记录值。第 7 次后的 data 步循环，第二个 set 语句不能读到据，no 变量值不被更新，并保留到后续的 data 步 PDV 中。

第四章

1．data 步程序

```
data jansales;
input item  $10. amount;
datalines;
trucks 1382
vans 1235
sedans 2391
   ;
run;
```

不能正确读取内嵌数据行。因为变量 item 的长度固定为 10 个字节，而数据行对应列的宽度是不固定的，如果小于 10 个字节，则会将后面的数据读入，读满 10 个字节，如果大于 10 个字节，则会进行截断。采用格式修饰符“:”可以正确读取，即

```
data jansales;
input item : $10. amount;
datalines;
trucks 1382
vans 1235
sedans 2391
   ;
run;
```

2．由于数据行中的等号之前的变量名为中文，不能作为数据集变量名，而命名读取方式又要求 input 语句中的变量名和数据行中的变量名相同，因此不能直接读取。

首先用 data 步把包含等号的数据全部作为字符读入

```
data worker;
input id $ sex $ name $10;
datalines;
054 性别=男 姓名=张三
069 性别=女 姓名=李英
;
run;
```

得出数据集 worker 为

	id	sex	name
1	054	性别=男	姓名=张三
2	069	性别=女	姓名=李英

然后采用字符截取函数进行截取

```
data worker1;
set worker;
sex=substr(sex,6,2);
name=substr(name,6,4);
run;
```

要注意，字符等号“=”的长度为 1 个字节，每个汉字的长度为 2 个字节。

3．采用列表方式读取，程序代码为

```
data temps;
input region $ date mmddyy10. +1 time time5. +1 tempf;
format date yymmdd10. time time8.;
datalines;
杨浦 10/10/2019   8:00 18  16:00  28  24:00  15
松江 10/10/2019   8:00 16  16:00  27  24:00  13
;
```

与格式输入方式的主要区别在于 input 语句读取完一个变量后数据指针的列位置。

4．将 cs300_2017 放入 d:\mydata 目录内，生成的临时文件 temp.txt 也放入该目录。

（1）data 步代码为

```
data _null_;
infile "d:\mydata\cs300_2017.txt";
input @3 stkcd $6.;
file "d:\mydata\temp.txt";
put "'"stkcd "'";
```

```
run;
```

data 语句中的自动变量_null_文件名，使 data 步不生产新数据集。不同次 data 步内置循环中 put 语句输出的 stkcd 的值位于目标文件 temp.txt 不同行，即换行输出。

（2）data 步代码为

```
data _null_;
infile "d:\mydata\cs300_2017.txt" end=js;
input @3 stkcd $6.;
file "d:\mydata\temp.txt";
if _n_=1 then put "where stkcd in (" @@;
put "'"stkcd "'" ","@@;
if js then put "'"stkcd "'" ");"@@ ;
run;
```

其中 put 语句选项@@使得 stkcd 的值在目标文件不换行输出，infile 语句选项定义变量 js，用于标识 input 语句读取的当前记录行是否最优一行。

这样得到的文件 temp.txt 中的内容是一个完整的 where 语句。

5．程序 4-45 代码为

```
data score13;
length mname $ 32;
do mname = 'score1.txt', 'score3.txt';
infile 'e:\lec' memvar=mname end=js firstobs=2;
do while (^js);
input name $ cal alg;
output;
end;
end;
stop;
run;
```

infile 语句选项 memvar=mname 导致 data 步读取所有文件后不会结束，而是继续下一次循环，因此需要 stop 语句避免死循环发生。

程序 4-46 代码为

```
data score123;
length memname $ 1024;
memname =' ';
infile 'e:\lec' memvar=memname end=wc firstobs=2;
put memname=;
do while(^wc);
input name $ cal alg ;
output;
end;
run;
```

infile 选项 memvar=memname 中的变量 memname 是空格，在读取最后一个文件后 DATA 会自动结束，因此不需要 stop 语句。

6．程序代码为

```
data A;
length vname $ 253;
vname=' ';
infile 'd:\mydata' memvar=vname;
m=vname;
input;
run;
```

需要注意，目标目录（本题为'd:\mydata'）内不能再有子目录，否则会出现错误。

7．运行

```
data s;
x=2;y=3;
set sashelp.class;
array nc[*] _numeric_;
put nc[5];
run;
```

信息窗口输出

```
112.5
84
98
102.5
102.5
83
84.5
112.5
84
99.5
50.5
90
77
112
150
128
133
85
112
```

语句array nc[*] _numeric_;将data步内的所有数值型变量定义为数据nc，nc中的变量次序为x、y、age、height和weight，put语句的nc[5]是变量weight的值。

8．先用程序4-65成数据集want

```
data want;
set book.yield5;
array term[*] _:;
do i=1 to dim(term);
yield=term[i];
type=i;
```

```
output;
end;
keep year yield type;
run;
```

运行一下 data 步程序将数据集 want 还原为 original

```
data original;
array y[1295,2] _temporary_;
if _n_=1 then
do i=1 to nb;
set want nobs=nb point=i;
y[i,1]=year;y[i,2]=yield;
end;
do i=1 to nb/5;
year=y[i,1];
_1=y[(i-1)*5+1,2];
_2=y[(i-1)*5+2,2];
_3=y[(i-1)*5+3,2];
_4=y[(i-1)*5+4,2];
_5=y[(i-1)*5+5,2];
output;
end;
drop yield type;
run;
```

9．将程序-66 修改为

```
data mapayh(drop=i);
set payh;
array lagp[5] _temporary_;
lagp[1]=clpr;
lagp[2]=lag(clpr);
do i=3 to 5;
lagp[i]=lag(lagp(i-1));
end;
MA5=mean(of lagp[*]);
run;
```

10．从程序 4-70 代码

```
data test(keep=name1 n);
set sashelp.class(rename=(age=age1 name=name1)) nobs=js;
n=0;
do i=1 to js ;
set sashelp.class point=i;
if age>age1 then n+1;
end;
run;
```

看出，data 步有两个 set 语句，第一个 set 语句没有 point=选项，而带 point=选

项的第二个 set 语句位于 do-end 循环之中，data 步内置循环由第一个 set 语句决定，每次内置循环中 do-end 循环结束后遇到 data 步结束边界，进入下一次内置循环，第一个 set 语句记录指针下移，直到遇到文件结束标志。只要有一个 set 的记录指针到达文件结束标志，则 data 步结束。因此，尽管没有 stop 语句，data 也不会死循环。

11．修改后的程序为

```
data test(keep=name1 n);
set sashelp.class(rename=(age=age1 name=name1))
nobs=js curobs=dq;
n=0;
do i=dq+1 to js ;
set sashelp.class point=i;
if age>age1 then n+1;
end;
run;
```

两个关键点：a)在第一个 set 语句中添加选项 curobs=dq，定义变量 dq 记录当前被读取的观测序号;b)do 循环中的起始值为当前观测序号加 1—dq+1。

12．先将数据集 sashelp.class 按 sex 和 height 排序，得出数据集 c1

```
proc sort data=sashelp.class out=c1;
by sex height;
run;
```

然后计算男生个数和女生个数，并用 data 步接口例程 symputx()生成宏变量 f 和 mn。

```
data temp;
set c1;
by sex;
if first.sex then n=0;
n+1;
if last.sex then output;
run;
data _null_;
set temp;
if sex="男" then call symputx("nm",n);
if sex="女" then call symputx("nf",n);
run;
```

最好采用 data 步间数据传递，用宏代换在下面 data 中获取男生个数和女生个数，据此得到同性别 height 值的中间作为中位数。

```
data median;
set c1;
by sex height;
if first.sex then n=0;
n+1;
if n=(&nm)/2 then output;
if n=(&nf)/2 then output;
run;
```

13．采用数组方法时，可以在第 3 步用数组方法求出中位数。即

```
data _null_;
array male[&nm] _temporary_;
array female[&nf] _temporary_;
if _n_=1 then
do;
do i=1 to &nm;
set c1 point=i;
male[i]=height;
end;
do i=1 to &nf;
p=i+&nm;
set c1 point=p;
female[i]=height;
end;
end;
med_h_m=male[%eval(&nm/2)];
med_h_f=female[%eval(&nf/2)];
put med_h_m= med_h_f=;
run;
```

14． 首先用 data 步生成数据集 work.p2p，其中变量 date 为债券交易日期，并按日期进行排序。

```
data p2p;
set book.p2p;
date=datepart(time);
time=timepart(time);
format date yymmdd10.;
format time time.;
keep date time sellid buyid id;
run;
proc sort data=p2p;
by date;
run;
```

以卖出者在卖出日期后 5 天是否有再买债权的统计为例。

```
data buy;
set p2p(rename=(sellid=sellid1 buyid=buyid1 date=date1))
nobs=gs curobs=cn;
date=0;
do i=cn to gs while(date<date1+5);
set p2p point=i;
if sellid=buyid1 then output;
end;
run;
```

第五章

1. where 数据集选项可以用于过程步输出数据集，例如

```
proc means data=sashelp.class;
var age height;
output out=sgx(where=(age>12));
run;
```

2. a)采用 where 语句程序为

```
proc print data=sashelp.class;
where sex="男";
run;
proc print data=sashelp.class;
where sex="女";
run;
```

输出结果为

Obs	Name	Sex	Age	Height	Weight
1	阿尔弗雷德	男	14	69.0	112.5
5	亨利	男	14	63.5	102.5
6	詹姆斯	男	12	57.3	83.0
9	杰弗瑞	男	13	62.5	84.0
10	约翰	男	12	59.0	99.5
15	菲利普	男	16	72.0	150.0
16	罗伯特	男	12	64.8	128.0
17	罗纳德	男	15	67.0	133.0
18	托马斯	男	11	57.5	85.0
19	威廉	男	15	66.5	112.0

Obs	Name	Sex	Age	Height	Weight
2	爱丽丝	女	13	56.5	84.0
3	芭芭拉	女	13	65.3	98.0
4	凯露	女	14	62.8	102.5
7	简	女	12	59.8	84.5
8	雅妮特	女	15	62.5	112.5
11	乔伊斯	女	11	51.3	50.5
12	茱迪	女	14	64.3	90.0
13	罗伊斯	女	12	56.3	77.0
14	玛丽	女	15	66.5	112.0

b)采用 where 数据集选项的程序为

```
proc print data=sashelp.class(where=(sex="男"));
run;
proc print data=sashelp.class(where=(sex="女"));
run;
```

输出结果与 a)相同。

c)采用 by 语句需要首先按 sex 将 sashelp.class 排序，即

```
proc sort data=sashelp.class out=class;
by sex;
proc print data=class;
by sex;
run;
```

输出结果为

性别=男

Obs	Name	Age	Height	Weight
1	阿尔弗雷德	14	69.0	112.5
2	亨利	14	63.5	102.5
3	詹姆斯	12	57.3	83.0
4	杰弗瑞	13	62.5	84.0
5	约翰	12	59.0	99.5
6	菲利普	16	72.0	150.0
7	罗伯特	12	64.8	128.0
8	罗纳德	15	67.0	133.0
9	托马斯	11	57.5	85.0
10	威廉	15	66.5	112.0

性别=女

Obs	Name	Age	Height	Weight
11	爱丽丝	13	56.5	84.0
12	芭芭拉	13	65.3	98.0
13	凯露	14	62.8	102.5
14	简	12	59.8	84.5
15	雅妮特	15	62.5	112.5
16	乔伊斯	11	51.3	50.5
17	茱迪	14	64.3	90.0
18	罗伊斯	12	56.3	77.0
19	玛丽	15	66.5	112.0

程序的执行结果不同。此外，where 语句和数据集选项需要执行两次 proc print，数据集选项在观测读入内存时起到过滤作用。by 语句只需要执行一次 proc print。

3．一些过程步遇到 run 语句并不推出运行，用于可以继续输入语句并通过提交 run 语句多次运行程序段，实现用户和程序的交互，以 run 语句为最后一条语句的语句组称为 run-组。

并不是所有程序都是交互运行方式，非交互运行方式的过程中只有一个 run 语句，是过程步边界，交互运行方式的过程需要用 quit 语句推出运行，如果不执行 quit 语句，过程仍然处于运行状态，一些输出数据集将不能生成。

4．程序代码为

```
proc transpose data=book.fishdata
out=trans(rename=(col1=measurement) drop=col2 col3);
var length1-length3;
by location;
run;
```

5．（1）一下程序生成日内 5 分钟、10 分钟、30 分钟、1 小时和日交易数据集_5m、_10m、_30m、_1h 和 day。

```
data _5m;
set book.zgsh;
if mod(minute(ttime),5)=0;
run;
data _10m;
set book.zgsh;
if mod(minute(ttime),10)=0;
run;
data _30m;
set book.zgsh;
if mod(minute(ttime),30)=0;
```

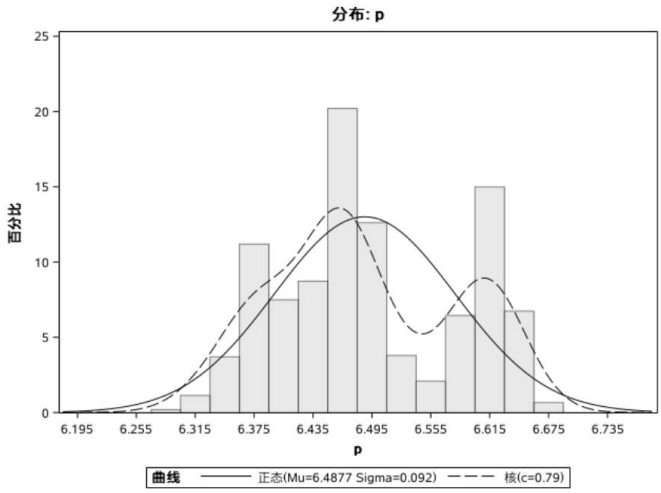
```
run;
data _1h;
set book.zgsh;
if minute(ttime)=0;
run;
data day;
set book.zgsh;
if hour(ttime)=15;
run;
```

(2)+（3）（原假设为正态分布）

（a）1 分钟交易价格频率直方图和分布检验结果

“正态”分布的拟合优度检验

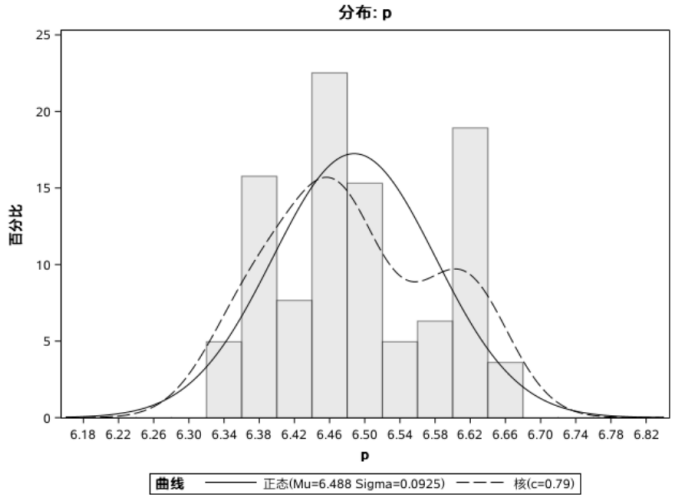
检验		统计量		p 值
Kolmogorov-Smirnov	D	0.1319104	Pr > D	<0.010
Cramer-von Mises	W-Sq	3.6102481	Pr > W-Sq	<0.005
Anderson-Darling	A-Sq	22.4504986	Pr > A-Sq	<0.005



（b）5 分钟交易价格频率直方图

“正态”分布的拟合优度检验

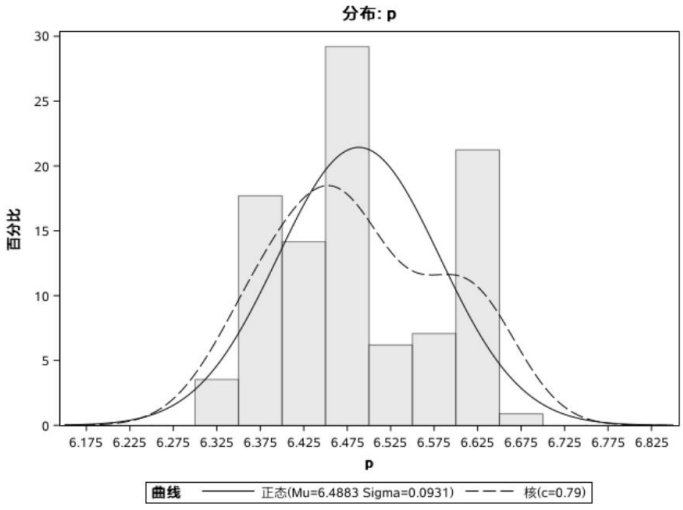
检验		统计量		p 值
Kolmogorov-Smirnov	D	0.13109493	Pr > D	<0.010
Cramer-von Mises	W-Sq	0.73146848	Pr > W-Sq	<0.005
Anderson-Darling	A-Sq	4.60557545	Pr > A-Sq	<0.005



（c）10 分钟交易价格频率直方图

“正态”分布的拟合优度检验

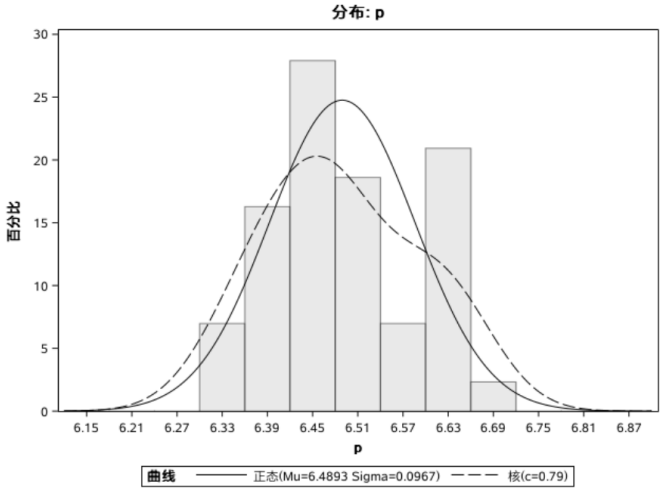
检验		统计量		p 值
Kolmogorov-Smirnov	D	0.18824858	Pr > D	0.050
Cramer-von Mises	W-Sq	0.10307886	Pr > W-Sq	0.097
Anderson-Darling	A-Sq	0.62939136	Pr > A-Sq	0.090



（d）30 分钟交易价格频率直方图

“正态”分布的拟合优度检验

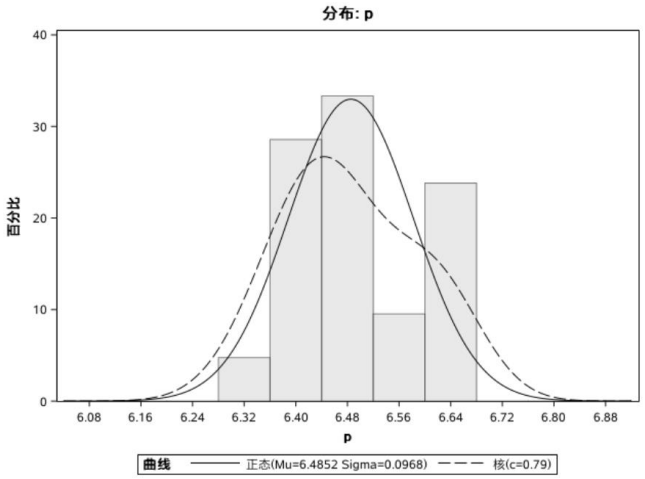
检验		统计量		p 值
Kolmogorov-Smirnov	D	0.18824858	Pr > D	0.050
Cramer-von Mises	W-Sq	0.10307886	Pr > W-Sq	0.097
Anderson-Darling	A-Sq	0.62939136	Pr > A-Sq	0.090



（e）1 小时交易价格频率直方图

“正态”分布的拟合优度检验

检验		统计量		p 值
Kolmogorov-Smirnov	D	0.18824858	Pr > D	0.050
Cramer-von Mises	W-Sq	0.10307886	Pr > W-Sq	0.097
Anderson-Darling	A-Sq	0.62939136	Pr > A-Sq	0.090



只有 4 个交易日，日频率直方图没有参考价值。

从图形看，随着抽样频率的降低，交易价格 p 的分布越来越靠近正态分布。从“正态”分布的拟合优度检验结果看，1 分钟和 5 分钟交易价格 p 的分布检验拒绝原假设，即不服从正态分布，10 分钟和 30 分钟和 1 小时交易价格的分布检验中，5%水平下拒绝原假设，但在 10%水平下不能拒绝原假设。

6．首先执行程序 5-32 生成 SAS 数据集 economics、finance 和 math

```
filename flist pipe 'dir d:\mydata\course*.xlsx/b';
data course;
length lec $ 60;
infile flist truncover length=l;
input majorcourse $60. @;
```

```
input @1 @"_" name1 $60. ;
lec='d:\mydata\'||majorcourse;
l=index(name1,'.')-1;
name=substr(name1,1,l);
put l name;
run;
data _null_;
set course;
txt1=cats("proc import
datafile='d:\mydata\",majorcourse,"'");
txt2=cats("out=",name);
txt3="dbms=excel replace;getnames=yes; run;";
txt=txt1||" "||txt2||" "||txt3;
call execute(txt);
run;
```

然后在 data 步中调用例程 execute()打印三个 SAS 数据集

```
data _null_;
set course;
txt1=cats("proc print data=",name,";");
txt=cats(txt1,"run;");
call execute(txt);
run;
```

7．利用 6.中生成的单个数据集，在 data 步调用例程 execute()对三个数据集执行过程 proc means，即

```
data _null_;
set course;
txt1=cats("proc means data=",name,";");
txt2=cats("output out=",name||" max=",";","run;");
txt=txt1||txt2;
call execute(txt);
run;
```

需要注意的是 cats 函数连接字符时，将被连接的字符前后空格删除，因此如下语句

```
txt2=cats("output out=",name," max=",";","run;");
```

形成的字符串被execute()解释为如下语句

```
output out=namemax=;
```

出现错误，尽管字符常数" max="中"max"前有空格以便和name分割，但cats函数连接前将空格删除。因此争取的做法时用"||"将name和" max="连接为整体后，再用cats函数和其它文本连接。

第六章

1．将实际跨度谁给你为 1999-2098 的语句为

```
option yearcutoff=1999;
```

然后执行程序

```
data _null_;
a='01mar15'd;
b="01mar97"d;
put 'a=' a date10.;
put 'b=' b date10.;
run;
```

结果显示

a= 01MAR2015

b= 01MAR2097

即世纪部分按 20 年（21 世纪）计算。

2．有两种读取方法。第一种方法时先将只有年份的外部数据读为字符，然后与字符'01JAN'连接，最后用 input()函数把字符型数据转换为日期型。第二种方法时将年份按数值读给变量 year，然后用函数 mdy(1,1,year)形成日期数据。

两种方法的关键在于：任何一个 SAS 日期数据都应该包含年、月、日信息，不含月和日的日期默认 1 月 1 日。

3．函数 nwkdom()用于计算给定年份和月份内各周每天对应的日期。采用 data 步 put 语句可以显示 2028 年 6 月第 3 周星期 6 对应的日期：

```
data _null_;
w=nwkdom(3,7,6,2028);
put "2028年6月的第3周星期6是" w yymmdd10.;
run;
```

运行结果显示为

2028 年 6 月的第 3 周星期 6 是 2028-06-17.

4．（1）程序代码为

```
data t;
set book.trt_600085_1m;
h=hour(_col2);
m=minute(_col2);
s=second(_col2);
run;
```

（2）程序代码为

```
data t;
set book.trt_600085_1m;
h=hour(_col2);
m=minute(_col2);
s=second(_col2);
datetime=dhms();
format datetime datetime.;
run;
```

生成的数据即 t 的数据结构为

	_COL9	_COL10	_COL11	_COL12	h	m	s	datetime
1	25.681	147200	3780241	0.2732	6	12	41	22MAR21:06:12:41
2	25.6416	53500	1371824	-0.3114	6	12	41	22MAR21:06:12:41
3	25.6222	121400	3110540	-0.3124	6	12	41	22MAR21:06:12:41
4	25.5589	39500	1009576	0.0392	6	12	41	22MAR21:06:12:41
5	25.585	15700	401685	0.3132	6	12	41	22MAR21:06:12:41
6	25.6599	47200	1211145	0.1171	6	12	41	22MAR21:06:12:41
7	25.7192	116400	2993715	0.5458	6	12	41	22MAR21:06:12:41
8	25.7953	115600	2981937	0.0775	6	12	41	22MAR21:06:12:41
9	25.8672	92900	2403060	0.3487	6	12	41	22MAR21:06:12:41
10	25.9724	265200	6887873	0.4633	6	12	41	22MAR21:06:12:41

（3）程序代码为

```
data t;
set book.trt_600085_1m;
d=put(_col2,date9.);
t=put(_col3,time8.);
dt=cats(d,":",t);
datetime=input(dt,datetime17.);
run;
```

5．函数 yrdif()的第 3 个自变量选做“age”即可，例如

```
data _null_;
birthday='29Jun1964'd;
today='29aug2021'd;
myage=yrdif(birthday,today,'AGE');
put '我今年' myage '岁了';
run;
```

运行结果显示

```
我今年 57.167123288 岁了
```

6．函数 inttest()用来判断一个给定的间隔名称是否合法。例如

```
data _null_;
x='year2.15';
y='year.13';
valid1=inttest(x);
valid2=inttest(y);
put valid1;
put valid2;
run;
```

输出结果显示 valid1 的值为 1，表明’year2.15’是一个有效的时间区间名称；valid2 的值为 0，表明’year.13’是一个无效的时间区间名称。

7．函数 intnx(‘week.3’,’02apr21’,3)计算的而是从 2021 年 4 月 2 日开始，三周后所在周的第三天对应的日期，即 4 月 20 日。可运行一下程序验证

```
data _null_;
x=intnx('week.3','02apr21'd,3);
put x yymmdd10.;
run;
```

输出结果显示：2021-04-20

8．函数形式为

```
intnx(‘week.2’,date,0,’B’)
```

9．按正常的计算方法，intnx(‘week.3’,’01jun20’d,2)是 2020 年 6 月 1 日两周后

所在周的第三天对应的日期，即 6 月 16 号。但运行程序

```
data _null_;
x=intnx('week.3','01jun20'd,2);
put x yymmdd10.;
run;
```

显示结果为：2020-06-09，即 6 月 9 日。其原因在于两周后所在周的第三天距离起始日期超过三周，不符合要求，最后以两周后所在周的第三天的日期作为函数返回值。

10. DATA 步中的滞后函数 lag*n*(x)的返回值与通常意义上的滞后值并不一样。DATA 步中的 lag 函数会建立自己的队列（queue），队列的长度为滞后阶数 n，并把队列中的值初始化为缺失值。当 lag 函数被执行时，将其队列中位于顶部的值作为返回值，将该值从队列中移除，队列中其它值按顺序上移，把自变量当前值填入队列底部。因此 lag*n*(x)的前 n 个返回值为缺失值。

由于其特殊的设计，一些情况下使用滞后函数会得出意外的结果，例如在 if 条件语句中或者前后语句中使用 lag 函数。由于 if 语句中条件不成立时后续语句不被执行，lag 函数的第一次执行是 if 语句条件满足时，此时的 lag 函数（长度为 1）的对列中的值为缺失值。

11. 首先运行程序 3-66，生成数据集 test

```
data test;
input stkcd $ date yymmdd8. clp 6.2 @@;
format date yymmddn8.;
datalines;
   600013 20030310 5.23  600013 20030311 5.25
   600013 20030312 5.20  600013 20030313 5.40
   600013 20030314 5.10  600059 20030310 7.23
   600059 20030311 7.25  600059 20030312 7.20
   600059 20030313 7.40  600059 20030314 7.10
   600582 20030310 12.23 600582 20030311 12.25
   600582 20030312 12.20 600582 20030313 12.40
   600582 20030314 12.10
;
```

然后计算对数收益率

```
data r;
set test;
r=dif(log(clp));
run;
```

对股票代码差分，将差分值不等于 0 的对应观测删除。

```
data r1;
set r;
dif=dif(stkcd);
if dif^=0 then delete;
run;
```

股票代码变量 stkcd 为字符型变量，差分时会自动转换为数值型变量。

12. 先定义月变量 m，然后按月份进行排序，采用自动变量 last.m 将每个月最后一天的利率数据输出。程序代码为

```
data temp;
set book.rdr7;
m=month(date);
y=year(date);
run;
proc sort data=temp;
by y m;
run;
data want;
set temp;
by y m;
if last.m;
run;
```

13．程序代码为

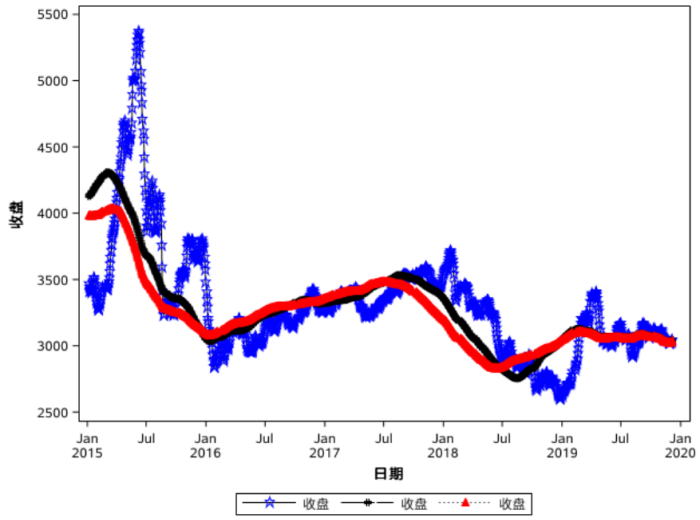
```
proc expand data=book.shd1519 out=ma;
convert clp=clp_ma5/transformout=(reverse movave 5
reverse);
convert clp=clp_ma120/transformout=(reverse movave 120
reverse);
convert clp=clp_ma180/transformout=(reverse movave 180
reverse);
run;
```

程序采用两个 reverse 关键词，第一个 reverse 将原序列倒序排列，然后计算后移平均，第二个 reverse 将算好的后移平均再倒排，回复原始时序。

14．程序代码为

```
proc sgplot data=ma;
series x=tdate y=clp_ma5/markers
markerattrs=(symbol=star size=2mm color=blue)
lineattrs=(thickness=1 pattern=solid);
series x=tdate y=clp_ma120/markers
markerattrs=(symbol=hash size=2mm)
lineattrs=(thickness=1 pattern=22);
series x=tdate y=clp_ma180/markers
markerattrs=(symbol=trianglefilled size=2mm color=red)
lineattrs=(thickness=1 pattern=dot);
run;
```

结果显示为



从时序图看出，计算窗口越宽，移动平均线越平滑。5 日移动平均线（蓝色五角星）ma5 变化最为剧烈，180 日移动均线（红色三角形）ma180 最为平缓。

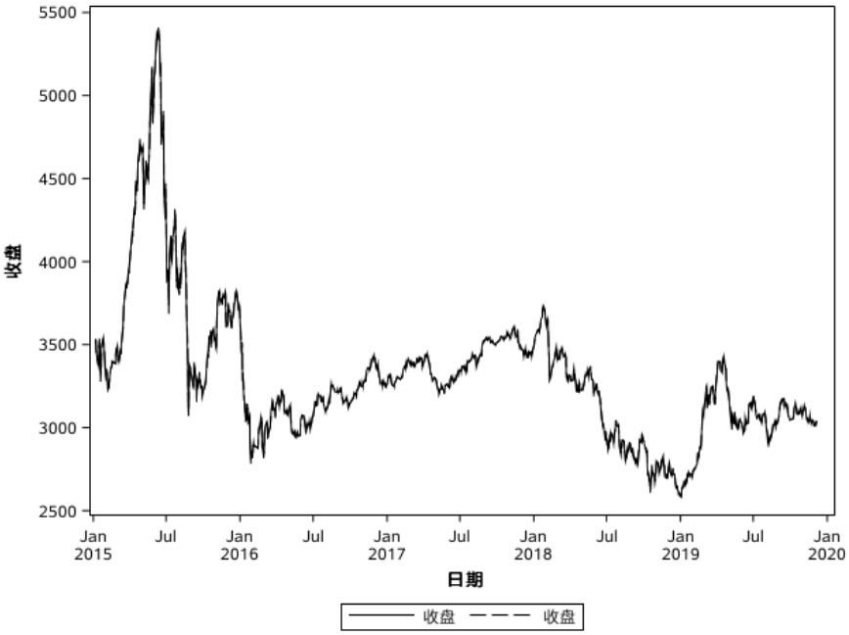
15．首先用 proc expand 生成指数平滑序列 clp_em96 和 clp_em75，程序代码为

```
proc expand data=book.shd1519 out=em;
convert clp=clp_em96/transformout=(ewma 0.96);
convert clp=clp_ma75/transformout=(ewma 0.75);
run;
```

然后用 proc sgplot 画出时序图，程序代码为

```
proc sgplot data=em;
series x=tdate y=clp_em96;
series x=tdate y=clp_em75;
run;
```

输出结果为



从结果看出，两个指数平滑序列区别不大，折线图几乎重合。

第七章

1．用 proc arima 进行单位根检验，程序代码为

```
proc arima data=book.tongrt;
identify var=clp_trt stationarity=(ADF=5);
run;
```

输出结果第一部分时序列的白噪声检验

白噪声的自相关检查

至滞后	卡方	自由度	Pr > 卡方	自相关					
6	6431.51	6	<.0001	0.981	0.967	0.950	0.936	0.921	0.910
12	9999.99	12	<.0001	0.896	0.886	0.875	0.863	0.854	0.842
18	9999.99	18	<.0001	0.832	0.825	0.817	0.808	0.804	0.798
24	9999.99	24	<.0001	0.794	0.788	0.781	0.774	0.768	0.761

表明序列 clp_trt 不是白噪声

第二部分为单位根检验，分三种检验模型列示检验结果。

第一种：检验模型没有常数项，也没有时间趋势项

增广 Dickey-Fuller 单位根检验

类型	滞后	Rho	Pr < Rho	Tau	Pr < Tau	F	Pr > F
零均值	0	-0.6775	0.5330	-1.06	0.2630		
	1	-0.6426	0.5401	-1.11	0.2413		
	2	-0.7293	0.5227	-1.32	0.1743		
	3	-0.6449	0.5396	-1.20	0.2118		
	4	-0.7052	0.5274	-1.30	0.1795		
	5	-0.6347	0.5417	-1.35	0.1650		

检验结果表明，无论检验模型的之后阶数是多少，都不能拒绝原假设，表明序列非平稳。

第二种：检验模型有常数项，没有时间趋势项

单均值	0	-22.0386	0.0072	-3.59	0.0064	6.67	0.0010
	1	-16.2572	0.0291	-2.91	0.0452	4.57	0.0512
	2	-16.4077	0.0280	-3.03	0.0329	5.11	0.0335
	3	-14.8254	0.0414	-2.80	0.0598	4.33	0.0660
	4	-16.3072	0.0287	-3.02	0.0342	5.05	0.0353
	5	-12.9206	0.0663	-2.75	0.0674	4.35	0.0648

检验结果表明，检验模型阶数为 2、3 和 5 时，5%水平上不能拒绝原假设，而其它阶数检验模型检验结果 5%水平上拒绝原假设。

第三种：检验模型有常数项，也有时间趋势项

趋势	0	-27.1623	0.0150	-3.87	0.0135	7.58	0.0159
	1	-21.1326	0.0547	-3.31	0.0657	5.47	0.0928
	2	-21.5089	0.0506	-3.47	0.0434	6.02	0.0626
	3	-21.3227	0.0526	-3.50	0.0399	6.28	0.0493
	4	-23.2510	0.0350	-3.74	0.0206	7.10	0.0266
	5	-19.1210	0.0826	-3.52	0.0378	6.42	0.0455

结果表明，检验模型阶数为 1 和 2 时 5%水平下不能拒绝原假设，其它阶数检验模型检验结果 5%水平下拒绝原假设。

2．（1）where 语句的作用时挑选复合条件的观测（工商银行）参与操作，单位根检验语句仍 是 identify 语句加选项 stationarity：

```
proc arima data=book.sh_bank;
where stkcd=601398;
identify var=clp stationarity=(PP=(1,2,3,4));
run;
```

输出结果为

Phillips-Perron 单位根检验

类型	滞后	Rho	Pr < Rho	Tau	Pr < Tau
零均值	1	0.1021	0.7069	0.23	0.7525
	2	0.0978	0.7059	0.21	0.7485
	3	0.0942	0.7050	0.20	0.7452
	4	0.0972	0.7057	0.21	0.7479
单均值	1	-7.1085	0.2683	-1.95	0.3091
	2	-7.4046	0.2502	-1.99	0.2922
	3	-7.6548	0.2359	-2.02	0.2786
	4	-7.4744	0.2462	-2.00	0.2884
趋势	1	-7.1636	0.6506	-1.78	0.7160
	2	-7.5249	0.6216	-1.83	0.6920
	3	-7.8302	0.5973	-1.87	0.6714
	4	-7.6096	0.6148	-1.84	0.6863

从结果看出，各种情况下都不能拒绝原假设，表明工商银行股票收盘价序列 clp 存在单位根，为非平稳序列。

（2）先计算收益率序列，data 步程序为

```
data return;
set book.sh_bank;
where stkcd=601398;
r=dif(log(clp));
```

```
run;
```

然后对变量 r 进行 ADF 检验

```
proc arima;
identify var=r stationarity=(adf=4);
run;
```

结果显示

增广 Dickey-Fuller 单位根检验

类型	滞后	Rho	Pr < Rho	Tau	Pr < Tau	F	Pr > F
零均值	0	-1144.62	0.0001	-33.14	<.0001		
	1	-1063.48	0.0001	-23.14	<.0001		
	2	-991.504	0.0001	-18.66	<.0001		
	3	-1683.02	0.0001	-18.46	<.0001		
	4	-3436.20	0.0001	-17.73	<.0001		
单均值	0	-1144.80	0.0001	-33.13	<.0001	548.86	0.0010
	1	-1063.99	0.0001	-23.13	<.0001	267.65	0.0010
	2	-992.433	0.0001	-18.66	<.0001	174.17	0.0010
	3	-1685.48	0.0001	-18.45	<.0001	170.27	0.0010
	4	-3448.76	0.0001	-17.73	<.0001	157.16	0.0010
趋势	0	-1145.54	0.0001	-33.14	<.0001	548.98	0.0010
	1	-1065.68	0.0001	-23.14	<.0001	267.75	0.0010
	2	-995.767	0.0001	-18.67	<.0001	174.37	0.0010
	3	-1698.69	0.0001	-18.47	<.0001	170.59	0.0010
	4	-3540.02	0.0001	-17.76	<.0001	157.73	0.0010

检验结果表明，各种情况下都在 1%显著水平下拒绝原假设，表明收益序列不存在单位根，为平稳序列。

3．采用 proc arima 的 identify 语句选项 ESCF 确定 ARMA 模型阶数，代码为

```
proc arima data=book.tongrt;
identify var=clp_trt ESACF;
run;
```

结果显示 0-5 阶的 ESACF 值概率值列表为

ESACF 概率值

滞后	MA 0	MA 1	MA 2	MA 3	MA 4	MA 5
AR 0	<.0001	<.0001	<.0001	<.0001	<.0001	<.0001
AR 1	0.0062	0.6558	0.0470	0.3360	0.0116	0.0907
AR 2	<.0001	0.5688	0.0648	0.5300	0.2434	0.9224
AR 3	<.0001	<.0001	0.4386	0.9250	0.2046	0.7517
AR 4	<.0001	<.0001	0.0012	0.6751	0.6628	0.3267
AR 5	<.0001	<.0001	<.0001	0.0331	0.6718	0.4302

尝试性阶数确定为

ARMA(p+d,q)
试验性阶选择检验

ESACF

p+d	q
2	1
1	5

(5% 显著性水平)

即 ARMA(2,1)和 ARMA（1,5）.确定阶数不唯一，更愿意采用 ARMA(2,1)模型，其包含的参数较少，更为简洁。

如果要建立 AR 模型，则需要在 identify 语句中将 MA 阶数设定为 0，即

```
proc arima data=book.tongrt;
identify var=clp_trt ESACF q=(0:0);
run;
```

结果显示

ESACF 概率值

滞后	MA 0
AR 0	<.0001
AR 1	0.0062
AR 2	<.0001
AR 3	<.0001
AR 4	<.0001
AR 5	<.0001

采用 AR（1）模型。模型阶数并不比 ARMA 模型阶数更高。从理论上讲，如果要充分捕捉序列中的自相关性，则 AR 模型阶数一般大于 ARMA 模型中的自回归阶数。但由于计算误差，往往实际中得出的实验性阶数并不遵循这种规律。

4．首先对 clp_trt 及其一阶差分进行单位根检验

```
proc arima data=book.tongrt;
identify var=clp_trt stationarity=(adf);
identify var=clp_trt(1) stationarity=(adf);
run;
```

检验结果表明，序列 clp_trt 不能确定其平稳性，因为不同检验模型在不同阶数下的检验结果不一致。序列 clp_trt 的一阶差分为平稳序列。

（1）对 clp_trt 的一阶差分建立模型，采用 MINIC 确定模型阶数，

```
proc arima data=book.tongrt;
identify var=clp_trt(1) minic;
run;
```

输出结果为

最小信息准则

滞后	MA 0	MA 1	MA 2	MA 3	MA 4	MA 5
AR 0	-1.26581	-1.3336	-1.32897	-1.35485	-1.34254	-1.35114
AR 1	-1.27181	-1.32762	-1.32664	-1.35107	-1.33776	-1.34584
AR 2	-1.28505	-1.3375	-1.32576	-1.34654	-1.33328	-1.33913
AR 3	-1.32141	-1.35983	-1.34494	-1.3399	-1.32658	-1.33315
AR 4	-1.31795	-1.35346	-1.33846	-1.33337	-1.32144	-1.32649
AR 5	-1.33074	-1.35597	-1.34025	-1.33486	-1.32206	-1.31983

误差序列模型: AR(10)

最小表值: BIC(3,1) = -1.35983

确定模型阶数为 ARMA（3，1）.

采用极大似然方法估计模型系数：

```
estimate p=3 q=1 method=ml;
run;
```

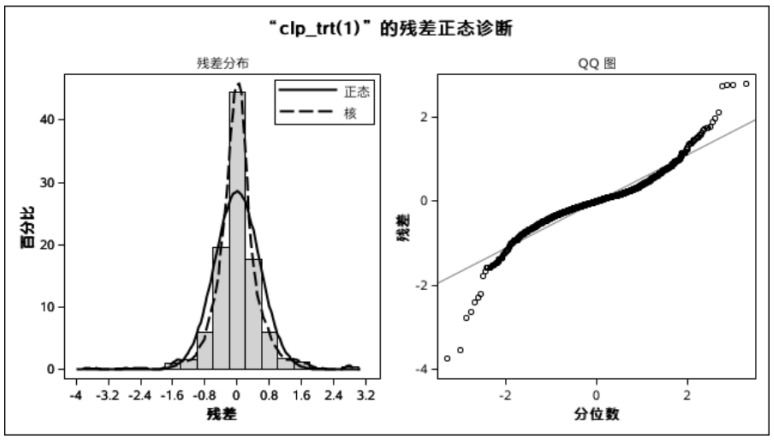
参数估计结果为

最大似然估计

参数	估计	标准误差	t 值	近似 Pr > |t|	滞后
MU	-0.01202	0.01541	-0.78	0.4354	0
MA1,1	0.15614	0.56076	0.28	0.7807	1
AR1,1	0.07034	0.55977	0.13	0.9000	1
AR1,2	-0.0072682	0.05666	-0.13	0.8979	2
AR1,3	0.05403	0.03071	1.76	0.0785	3

其中 mu 为模型均值，MA1,1 为移动平均项系数，AR1,1-AR1,3 为自回归项系数。从估计结果看出,所有系数的 t 检验都不显著,表明 clp_trt 的一阶差分是一个白噪声。

（2）残差分布诊断图为



从中看出，残差的分布具有明显的尖峰厚尾特征，无论是其频率直方图还是 QQ 图都很

明显地显示出这一特点。

残差不服从正态分布时，采用正态分布假设进行的极大似然估计称为拟极大似然（QML：Quasi-ML），不影响参数估计的一致性和渐近正态性，估计结果仍然可信，但会影响参数估计渐近分布的协方差计算，因此对应的 t-检验不可信。

（3）outmodel=选项将模型参数估计结果保存为 SAS 数据集，内容包括参数估计值、参数估计标准差和其它信息。outstat=选项将模型诊断统计值保存为 SAS 数据集，内容包含估计方法、诊断统计量等。

将 clp_trt 一阶差分的 ARMA（3，1）模型估计结果和诊断结果保存为 SAS 数据集的程序代码为

```
estimate p=3 q=1 method=ml outmodel=m outstat=s;
run;
```

（4）预测代码为（在 proc 语句中添加选项 plots=）

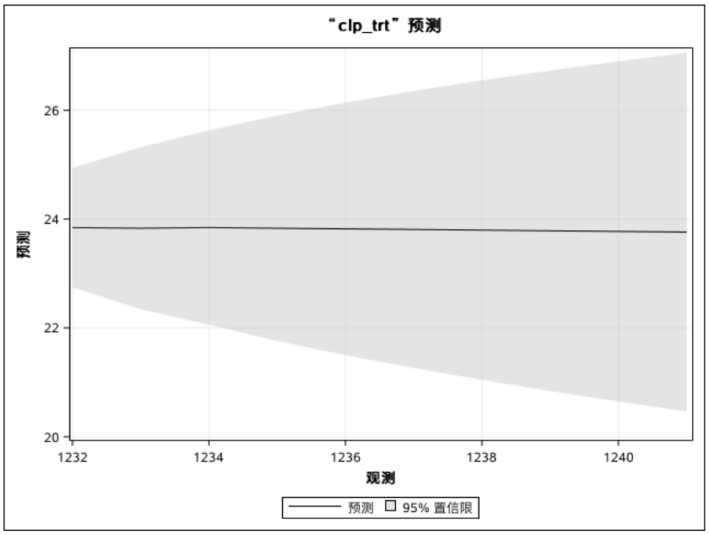
```
proc arima data=book.tongrt plots=forecast(all);
identify var=clp_trt(1) minic;
run;
estimate p=3 q=1 method=ml outmodel=m outstat=s;
run;
forecast lead=10 out=f;
quit;
```

10 步预测的预测值为

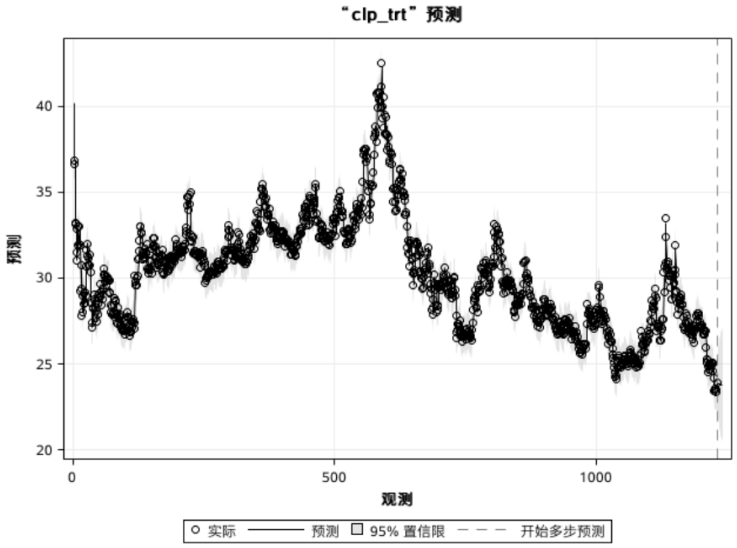
变量“clp_trt”的预测

观测	预测	标准误差	95% 置信限	
1232	23.8434	0.5596	22.7465	24.9403
1233	23.8322	0.7583	22.3460	25.3184
1234	23.8444	0.9106	22.0597	25.6291
1235	23.8317	1.0557	21.7625	25.9009
1236	23.8195	1.1830	21.5008	26.1382
1237	23.8088	1.2976	21.2655	26.3520
1238	23.7968	1.4035	21.0460	26.5476
1239	23.7848	1.5019	20.8410	26.7285
1240	23.7728	1.5943	20.6480	26.8976
1241	23.7608	1.6816	20.4648	27.0568

样本外预测值及其置信区间图形为



原始数据和预测数据的时序图为



竖虚线为分割线，作为样本内实际值和样本外预测值图形的分割线。

尽管建立的是 clp_trt 一阶差分的模型，预测值是原始序列 clp_trt 的预测值。

5. proc arima 是建立一元时间序列的自回归分布滞后模型，proc autoreg 是建立时间序列变量间的回归模型，模型误差项具有自相关性。将 proc autoreg 中语句 model 的自变量设置为因变量的滞后变量可以建立自回归模型，也可以用 model=/lag=建立自回归模型，实现 arima 的功能。proc autoreg 与 proc arima 的另一个显著区别在于，前者可以建立二阶矩模型——误差项的条件异方差 GARCH 模型，后者则无此功能。

6. （1）先用 data 步生日收益率变量 r，为了避免跨股票处观测生成的收益率，对 stkcd 进行一阶差分，并将一阶差分不等于 0 的观测删除。同时把 r 为缺失值的观测删除

```
data r_yy;
set book.sh_yiyao;
r=dif(log(clp));
if not dif(stkcd);
if r;
run;
```

（2）程序代码为

```
data fxyy;
set r_yy;
where stkcd=600196;
run;
```

（3）KPSS 检验只能用 proc autoreg 进行，proc arima 无此功能。

对收盘价 clp 的 kpss 检验

```
proc autoreg data=fxyy;
model clp=/stationarity=(kpss);
run;
```

检验结果显示为

KPSS 平稳性检验

类型	滞后	Eta	Pr > Eta
单均值	22	1.5832	0.0002
趋势	22	0.4703	0.0002

注意 KPSS 检验的原假设是没有单位根，拒绝原假设意味着 clp 存在单位根，为非平稳序列。

对日收益率 r 的 kpss 检验

```
proc autoreg data=fxyy;
model r=/stationarity=(kpss);
run;
```

检验结果显示为

KPSS 平稳性检验

类型	滞后	Eta	Pr > Eta
单均值	22	0.0936	0.6179
趋势	22	0.0773	0.2877

不能拒绝原假设，意味着日收益率为平稳时间序列。

需要注意：proc autoreg 没有交互运行方式，不能出现多个 run 组，在同一过程中运行多个 model 语句，因此对 clp 及其一阶差分的 KPSS 检验要在两个 proc autoreg 过程中独立进行。

KPSS 检验以不存在单位根为原假设，区别于其它检验方法以存在单位根为原假设。

（4）用 proc arima 确定复星医药的日收益率 r 的 AR 阶数，程序代码为

```
proc arima data=fxyy;
identify var=r minic q=(0:0);
identify var=r esacf q=(0:0);
run;
```

最小信息准则结果为

最小信息准则	
滞后	MA 0
AR 0	-7.33781
AR 1	-7.33288
AR 2	-7.3306
AR 3	-7.32683
AR 4	-7.3224
AR 5	-7.3181

误差序列模型: AR(5)

最小表值: BIC(0,0) = -7.33781

扩展偏自相关函数 ESACF 定阶结果为

ESACF 概率值	
滞后	MA 0
AR 0	0.2370
AR 1	<.0001
AR 2	<.0001
AR 3	<.0001
AR 4	<.0001
AR 5	<.0001

ARMA(p+d,q) 试验性阶选择检验	
ESACF	
p+d	q
0	0

(5% 显著性水平)

两种方法定阶结果都是 AR（0），表明 r 为白噪声。

（5）为了演示 proc autoreg 的用法，对 r 建立 AR(1)模型。两种建立方法

a）采用 r 的一阶滞后作为解释变量。

首先生成包含 r 一阶滞后变量的数据集 fxyy1

```
data fxyy1;
set fxyy;
r_1=lag(r);
run;
```

然后在 proc autoreg 的 model 语句中将 r_1 作为解释变量，即

```
proc autoreg data=fxyy1;
```

```
model r=r_1;
run;
```

参数估计结果为

参数估计

变量	自由度	估计	标准误差	t 值	近似 Pr > |t|
Intercept	1	0.000793	0.000744	1.07	0.2867
r_1	1	-0.0344	0.0290	-1.18	0.2370

b）在 model 语句中添加选项 nlag=，即

```
proc autoreg data=fxyy1;
model r=/nlag=1;
run;
```

估计结果显示为

自回归参数的估计

滞后	系数	标准误差	t 值
1	0.034279	0.028996	1.18

Yule-Walker 估计

SSE	0.78139256	DFE	1188
MSE	0.0006577	均方根误差	0.02565
SBC	-5329.5413	AIC	-5339.7047
MAE	0.01840277	AICC	-5339.6946
MAPE	104.671115	HQC	-5335.8746
Durbin-Watson	1.9937	转换回归 R 方	0.0000
		总 R 方	0.0012

参数估计

变量	自由度	估计	标准误差	t 值	近似 Pr > |t|
Intercept	1	0.000771	0.000719	1.07	0.2834

两种估计方法的结果很接近。

（6）首先用 data 步生成复星医药日交易量对数变量 lvol

```
data fxyy;
set book.fxyy;
r=dif(log(clp));
lvol=log(vol);
```

```
run;
```

然后用 proc autoreg 对 r 建立 AR(1)+GARCH(1,1)，并且在 GARCH 模型中引入外生变量 lvol

```
proc autoreg data=fxyy;
model r=/nlag=1 garch=(p=1,q=1);
hetero lvol;
run;
```

参数估计结果显示为

参数估计

变量	自由度	估计	标准误差	t 值	近似 Pr > |t|
Intercept	1	0.000918	0.000555	1.65	0.0984
AR1	1	0.0287	0.0298	0.96	0.3357
ARCH0	1	1.0537E-8	0	Infty	<.0001
ARCH1	1	0.0541	0.008250	6.56	<.0001
GARCH1	1	0.9439	0.008021	117.67	<.0001
HET1	1	1.5239E-7	6.7244E-8	2.27	0.0234

参数估计结果表明，尽管 r 本身不存在自相关，自回归系数的估计结果不显著，但存在明显的 GARCH 现象，GARCH 模型中参数估计均显著。GARCH 模型中外生变量 lvol 系数显著不为 0，表明交易量对日收益率波动具有显著影响。

7．（1）程序代码为

```
data sh;
set book.shindex;
rename clp=clp_shindex;
run;
data bys;
set book.bys;
rename clp=clp_bys;
run;
```

（2）程序代码为

```
proc sort data=sh;
by tdate;
proc sort data=bys;
by tdate;
run;
```

（3）程序代码为

```
data shid_bys;
merge sh bys;
by tdate;
run;
```

（4）程序代码为

```
data shid_bys;
set shid_bys;
r_shind=dif(log(clp_shindex));
r_b=dif(log(clp_bys));
run;
```

（5）程序代码为

```
data shid_bys;
set shid_bys;
r_shind_1=lag(r_shind);
r_shind_2=lag2(r_shind);
r_b_1=lag(r_b);
r_b_2=lag2(r_b);
run;
```

（6）采用滞后变量作为自变量的方法，程序代码为

```
proc autoreg data=shid_bys;
model r_b=r_b_1 r_b_2 r_shind r_shind_2;
run;
```

采用 nlag=选项的方法程序代码为

```
proc autoreg data=shid_bys;
model r_b=r_shind r_shind_2/nlag=2;
run;
```

（7）EGARCH 模型中杠杆效应参数的表示用自动变量_THETA_，此时要求 model 语句中有 GARCH=(type=EGARCH,p=1,q=1)选项。程序代码为

```
proc autoreg data=shid_bys;
model r_bys=r_shind r_shind_2/nlag=2
garch=(type=EGARCH,p=1,q=1);
restrict _THETA_=0;
run;
```

（8）用 test 语句检验杠杆效应参数是否为 0，程序代码为

```
proc autoreg data=shid_bys;
model r_bys=r_shind r_shind_2/nlag=2
garch=(type=EGARCH,p=1,q=1);
test _THETA_=0;
run;
```

参数估计和约束条件检验输出显示为

检验 1

源	自由度	F 值	Pr > F
分子	1	14.43	0.0002
分母	1086		

参数估计

变量	自由度	估计	标准误差	t 值	近似 Pr > |t|
Intercept	1	0.0000919	0.000463	0.20	0.8426
r_shind	1	1.0523	0.0387	27.18	<.0001
r_shind_2	1	-0.0695	0.0425	-1.63	0.1024
AR1	1	0.1030	0.0316	3.26	0.0011
AR2	1	0.0738	0.0328	2.25	0.0245
EARCH0	1	-0.3986	0.0686	-5.81	<.0001
EARCH1	1	0.0595	0.0150	3.96	<.0001
EGARCH1	1	0.9507	0.008493	111.95	<.0001
THETA	1	1.6452	0.4331	3.80	0.0001

检验采用的是 F 检验，检验结果拒绝原假设，表明杠杆效应显著。

（9）在 model 语句添加选项 runs=(z=sr)可对模型估计后的标准化残差进行独立检验，程序代码为

```
proc autoreg data=shid_bys;
model r_bys=r_shind r_shind_2/nlag=2 runs=(z=sr)
garch=(type=EGARCH,p=1,q=1);
run;
```

输出结果显示

运行独立性检验

RUNS	Pr > |RUNS|
-0.9786	0.3278

表明不能拒绝原假设，即可以认为标准化残差为独立序列，模型设定合理。

（10） 在 model 语句添加选项 BP=(M=3)可对模型进行变点检验，程序代码为

```
proc autoreg data=shid_bys;
model r_bys=r_shind r_shind_2/nlag=2 BP=(m=3)
garch=(type=EGARCH,p=1,q=1);
run;
```

由于没有指明采用何种检验统计量，检验结果列出四种检验统计量及其检验结果

a) supF 检验

Bai-Perron 多结构变化检验

supF 检验

中断数	supF	Pr > supF
1	23.83	0.0011
2	20.28	0.0003
3	16.00	0.0023

表明变点个数至少为 3 个。

b) UDmaxF 检验

Bai-Perron 多结构变化检验

UDmaxF 检验

中断数	UDmaxF	Pr > UDmaxF
3	23.8316166	0.0012

表明变点个数至少为 3 个。

c) WDmaxF 检验

Bai-Perron 多结构变化检验

WDmaxF 检验

中断数	Alpha	WDmaxF	Pr > WDmaxF
3	0.1000	23.8316166	0.0012
	0.0500	23.8316166	0.0011
	0.0250	23.8316166	0.0014
	0.0100	23.9690527	0.0015

仍然支持变点个数至少为 3 个的结论。

d) supF（l+1|l）检验

Bai-Perron 多结构变化检验

supF(l+1|l) 检验

l	新断点	supF(l+1|l)	Pr > supF(l+1|l)
0	877	23.8316166	0.0041
1	938	10.3796919	0.6334
2	642	6.70634262	0.9810
3	706	8.70468051	0.8467

给出可能的变点。

第八章

1．（1）程序代码为

```
proc varmax data=book.sh_bys_zsyh_r;
model r_sh r_b r_z/minic=(p=3 q=2type=AIC);
run;
proc varmax data=book.sh_bys_zsyh_r;
model r_sh r_b r_z/minic=(p=3 q=2 type=AICC);
run;
proc varmax data=book.sh_bys_zsyh_r;
model r_sh r_b r_z/minic=(p=3 q=2 type=HQC);
run;
```

```
proc varmax data=book.sh_bys_zsyh_r;
model r_sh r_b r_z/minic=(p=3 q=2 type=SBC);
run;
```

proc varmax 没有交互运行模式，况且选项 minic=(type=value)只能给出一种信息准则，因此需要执行四个 proc varmax 来验证不同的信息准则确定的模型阶数。为了减少程序运行时间，这里将自回归和移动平均的最大阶数限定为 3 和 2.

为节省空间，这里给出以 HQC 为准则的定阶结果，显示为

基于"HQC"的最小信息准则

滞后	MA 0	MA 1	MA 2
AR 0	-25.59466	-25.61733	-25.62018
AR 1	-25.57959	-25.60051	-25.59832
AR 2	-25.56416	-25.59939	-25.58426
AR 3	-25.57187	-25.58143	-25.56952

SAS 系统

VARMAX 过程

模型类型	VMA(2)
估计方法	最大似然估计

显示处定阶结果为 VARMA(0,2)，即 2 阶向量移动平均过程 VMA(2)。

（2）在 model 中添加选项 trend=可以在模型中添加时间趋势。程序代码为

```
proc varmax data=book.sh_bys_zsyh_r;
model r_sh r_b r_z/q=2 p=0 trend=linear;
run;
```

（3）在 model 中添加选项 method=选择模型参数估计方法，method=cml 为条件极大似然估计。程序代码为

```
proc varmax data=book.sh_bys_zsyh_r;
model r_sh r_b r_z/q=2 p=0 trend=linear method=cml;
run;
```

（4）在 model 中添加选项 print=(diagnose)，会在结果输出中显示残差白噪声混成检验（Portmanteau Test），程序代码为

```
proc varmax data=book.sh_bys_zsyh_r;
model r_sh r_b r_z/q=2 p=0
trend=linear method=cml print=(diagonose);
  run;
```

结果显示为

残差互相关的 Portmanteau 检验			
至多滞后	自由度	卡方	Pr > 卡方
3	9	6.24	0.7156
4	18	21.30	0.2648
5	27	43.57	0.0229
6	36	58.69	0.0098
7	45	68.88	0.0125
8	54	82.07	0.0082
9	63	107.35	0.0004
10	72	109.46	0.0029
11	81	126.48	0.0009
12	90	132.50	0.0024

从结果看出，滞后 3 阶和 4 阶时的混成检验不能拒绝原假设，但超过 4 阶后均拒绝原假设。表明模型设定不充足（adequate），需要改进。

（5）test 语句进行检验，用 ltrend(i)表示第 i 个分量模型的线性趋势项系数。以检验第一个分量模型是否应该带时间趋势项为例，等价于检验 ltrend(1)是否为 0，程序代码为

```
proc varmax data=book.sh_bys_zsyh_r;
model r_sh r_b r_z/q=1 p=2
trend=linear method=cml print=(diagnose);
test ltrend(1);
run;
```

运行出现错误，信息窗口提示：

```
WARNING: The TEST statement job cannot be completed
since a matrix is not positive definite.
```

构造 wald 检验的矩阵不是正定的，因此 test 无法完成！

2. proc varmax 模型能计算单个时间序列的分数单整阶数，这个功能是 proc arima 和 proc autoreg 都没有的。

（1）程序代码为

```
data bank_js;
set book.sh_bank;
if stkcd=601939;
lnclp=log(clp);
if clp;
run;
```

（2）在 model 语句添加选项 fi，可以计算时间序列的单整阶数。程序代码为

```
proc varmax data=bank_js;
model clp/p=3 FI;
```

```
run;
proc varmax data=bank_js;
model lnclp/p=3 FI;
run;
```

计算分数单整阶数需要很大的计算量，程序运行时间较长。

第一个 proc varmax 运行时间 6 分多钟，结果中参数估计显示为

模型参数估计

方程	参数	估计	标准误差	t 值	Pr > |t|	变量
clp	CONST1	1.25840	0.36154	3.48	0.0005	1
	AR1_1_1	0.60296	0.03512	17.17	0.0001	clp(t-1)
	AR2_1_1	0.10600	0.03325	3.19	0.0015	clp(t-2)
	AR3_1_1	0.08857	0.02854	3.10	0.0020	clp(t-3)
	D1	0.48710	0.01917	25.41	0.0001	

其中 D1 为分数单整阶数，clp 的分数单整阶数为 0.48710。

第二个 proc varmax 运行时间 10 分 43 秒多钟，估计结果显示为

模型参数估计

方程	参数	估计	标准误差	t 值	Pr > |t|	变量
lnclp	CONST1	0.36411	0.06993	5.21	0.0001	1
	AR1_1_1	0.58411	0.03313	17.63	0.0001	lnclp(t-1)
	AR2_1_1	0.13122	0.03288	3.99	0.0001	lnclp(t-2)
	AR3_1_1	0.08404	0.02854	2.94	0.0033	lnclp(t-3)
	D1	0.48894	0.01562	31.30	0.0001	

lnclp 的分数单整阶数为 0.48894.

在分数单整序列中，当 d=D1<0.5 时，序列自相关函数的递减速度为

$$\rho(k)\sim ck^{2d-1}=c\frac{1}{k^{1-2d}},k=1,2,\ldots$$

d 越大递减速度越慢。表明 lnclp 的长记忆性要强于 clp。

（3）采用 proc expand 可将分数单整的序列按单整阶数进行分数差分，程序代码为

```
proc expand data=bank_js out=js_df;
convert clp=dfclp/transform=(fdif 0.48871);
run;
```

采用 proc arima 对 js_df 中变量 dfclp 进行单位根检验

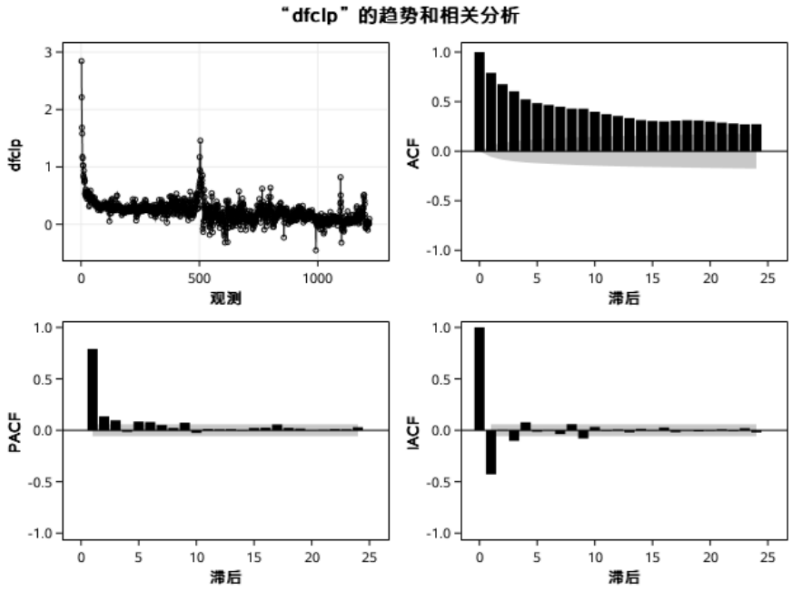
```
proc arima data=js_df;
identify var=dfclp stationarity=(ADF=(2));
run;
```

检验结果显示

增广 Dickey-Fuller 单位根检验

类型	滞后	Rho	Pr < Rho	Tau	Pr < Tau	F	Pr > F
零均值	2	-64.0335	<.0001	-7.09	<.0001		
单均值	2	-157.878	0.0001	-10.72	<.0001	57.70	0.0010
趋势	2	-226.347	0.0001	-12.42	<.0001	77.70	0.0010

表明原序列分数差分后的 fdclp 不存在单位根，为平稳序列。同时输出时序图、ACF 图和 PACF 图为



均显示处平稳序列的特征。

3. （1）程序代码为

```
data zjyy_216;
set book.sh_yiyao(rename=(clp=p_216));
if stkcd=600216;
keep tdate p_216;
run;
data lrzy_285;
set book.sh_yiyao(rename=(clp=p_285));
if stkcd=600285;
keep tdate p_285;
run;
data pzh_436;
set book.sh_yiyao(rename=(clp=p_436));
if stkcd=600436;
keep tdate p_436;
run;
data xhyl_587;
set book.sh_yiyao(rename=(clp=p_587));
if stkcd=600587;
```

```
keep tdate p_587;
run;
```

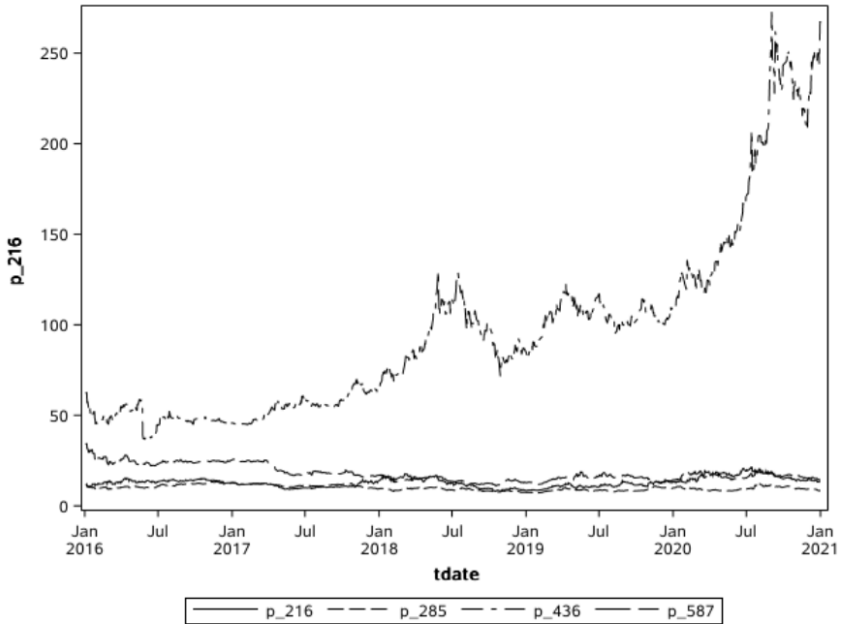
（2）程序代码为

```
proc sort data=zjyy_216;
by tdate;
proc sort data=lrzy_285;
by tdate;
proc sort data=pzh_436;
by tdate;
proc sort data= xhyl_587;
by tdate;
run;
data combine;
merge zjyy_216 lrzy_285 pzh_436 xhyl_587;
by tdate;
run;
```

（3）程序代码为

```
proc sgplot data=combine;
series x=tdate y=p_216;
series x=tdate y=p_285;
series x=tdate y=p_436;
series x=tdate y=p_587;
run;
```

图形显示为



（4）生成收益数据集

```
data combine_r;
set combine;
r_216=dif(log(p_216));
```

```
r_285=dif(log(p_285));
r_436=dif(log(p_436));
r_587=dif(log(p_587));
if r_216;
run;
```

if 语句删除有缺失值的观测。

检验各收益率序列平稳性程序代码为

```
proc arima data=combine_r;
identify var=r_216 stationarity=(ADF);
run;
identify var=r_285 stationarity=(ADF);
run;
identify var=r_436 stationarity=(ADF);
run;
identify var=r_587 stationarity=(ADF);
run;
quit;
```

单位根检验结果为

a）r_216 和 r_285

增广 Dickey-Fuller 单位根检验

类型	滞后	Rho	Pr < Rho	Tau	Pr < Tau	F	Pr > F
零均值	0	-1258.16	0.0001	-37.13	<.0001		
	1	-1268.57	0.0001	-25.18	<.0001		
	2	-1347.81	0.0001	-20.66	<.0001		
单均值	0	-1258.18	0.0001	-37.12	<.0001	688.84	0.0010
	1	-1268.61	0.0001	-25.17	<.0001	316.72	0.0010
	2	-1347.98	0.0001	-20.66	<.0001	213.36	0.0010
趋势	0	-1258.28	0.0001	-37.10	<.0001	688.36	0.0010
	1	-1268.92	0.0001	-25.16	<.0001	316.50	0.0010
	2	-1349.15	0.0001	-20.65	<.0001	213.31	0.0010

增广 Dickey-Fuller 单位根检验

类型	滞后	Rho	Pr < Rho	Tau	Pr < Tau	F	Pr > F
零均值	0	-1209.55	0.0001	-35.60	<.0001		
	1	-1218.50	0.0001	-24.66	<.0001		
	2	-1341.65	0.0001	-20.78	<.0001		
单均值	0	-1210.47	0.0001	-35.61	<.0001	634.00	0.0010
	1	-1221.50	0.0001	-24.68	<.0001	304.53	0.0010
	2	-1347.78	0.0001	-20.79	<.0001	216.20	0.0010
趋势	0	-1210.47	0.0001	-35.59	<.0001	633.45	0.0010
	1	-1221.51	0.0001	-24.67	<.0001	304.27	0.0010
	2	-1347.73	0.0001	-20.78	<.0001	216.02	0.0010

表明 r_216 和 r_285 均为平稳序列

b）r_436 和 r_587

增广 Dickey-Fuller 单位根检验

类型	滞后	Rho	Pr < Rho	Tau	Pr < Tau	F	Pr > F
零均值	0	-1176.12	0.0001	-34.53	<.0001		
	1	-1211.21	0.0001	-24.50	<.0001		
	2	-1122.29	0.0001	-19.46	<.0001		
单均值	0	-1177.95	0.0001	-34.57	<.0001	597.63	0.0010
	1	-1216.90	0.0001	-24.55	<.0001	301.33	0.0010
	2	-1133.69	0.0001	-19.53	<.0001	190.68	0.0010
趋势	0	-1180.45	0.0001	-34.64	<.0001	599.88	0.0010
	1	-1224.76	0.0001	-24.63	<.0001	303.22	0.0010
	2	-1146.64	0.0001	-19.59	<.0001	191.90	0.0010

增广 Dickey-Fuller 单位根检验

类型	滞后	Rho	Pr < Rho	Tau	Pr < Tau	F	Pr > F
零均值	0	-1101.16	0.0001	-32.50	<.0001		
	1	-1099.82	0.0001	-23.44	<.0001		
	2	-1106.35	0.0001	-19.39	<.0001		
单均值	0	-1102.30	0.0001	-32.52	<.0001	528.78	0.0010
	1	-1103.03	0.0001	-23.46	<.0001	275.19	0.0010
	2	-1111.61	0.0001	-19.41	<.0001	188.37	0.0010
趋势	0	-1102.94	0.0001	-32.52	<.0001	528.89	0.0010
	1	-1104.93	0.0001	-23.47	<.0001	275.44	0.0010
	2	-1114.02	0.0001	-19.41	<.0001	188.41	0.0010

结果表明 r_285 和 r_587 均为平稳序列。

（5）变量观测值有缺失值时，proc varmax 将无法执行，为此先将参与 VAR 模型的变量缺失值去掉，然后建立 VAR(2)模型。

程序代码为

```
data c_r;
set combine_r;
if r_216 & r_285 & r_436 & r_587;
run;
proc varmax data=c_r;
model r_216 r_285 r_436 r_587/p=2 method=ls;
run;
```

参数估计结果显示为

常数项和自回归系数矩阵

r_216	CONST1	0.00011	0.00081	0.14	0.8917	1
	AR1_1_1	-0.06447	0.03628	-1.78	0.0758	r_216(t-1)
	AR1_1_2	-0.01915	0.04777	-0.40	0.6885	r_285(t-1)
	AR1_1_3	0.07548	0.03597	2.10	0.0361	r_436(t-1)
	AR1_1_4	-0.08949	0.03990	-2.24	0.0251	r_587(t-1)
	AR2_1_1	-0.00222	0.03624	-0.06	0.9512	r_216(t-2)
	AR2_1_2	-0.06097	0.04775	-1.28	0.2020	r_285(t-2)
	AR2_1_3	-0.01046	0.03597	-0.29	0.7713	r_436(t-2)
	AR2_1_4	0.03601	0.04001	0.90	0.3684	r_587(t-2)
r_285	CONST2	-0.00076	0.00065	-1.17	0.2424	1
	AR1_2_1	-0.02494	0.02933	-0.85	0.3953	r_216(t-1)
	AR1_2_2	-0.06224	0.03862	-1.61	0.1074	r_285(t-1)
	AR1_2_3	0.04167	0.02908	1.43	0.1522	r_436(t-1)
	AR1_2_4	0.00816	0.03226	0.25	0.8004	r_587(t-1)
	AR2_2_1	0.00907	0.02930	0.31	0.7568	r_216(t-2)
	AR2_2_2	-0.01604	0.03861	-0.42	0.6779	r_285(t-2)
	AR2_2_3	0.02729	0.02908	0.94	0.3482	r_436(t-2)
	AR2_2_4	-0.02294	0.03235	-0.71	0.4783	r_587(t-2)
r_436	CONST3	0.00086	0.00079	1.08	0.2785	1
	AR1_3_1	-0.03874	0.03542	-1.09	0.2744	r_216(t-1)
	AR1_3_2	0.01101	0.04665	0.24	0.8135	r_285(t-1)
	AR1_3_3	0.01976	0.03512	0.56	0.5738	r_436(t-1)
	AR1_3_4	-0.09582	0.03896	-2.46	0.0141	r_587(t-1)
	AR2_3_1	0.00090	0.03539	0.03	0.9798	r_216(t-2)
	AR2_3_2	-0.01684	0.04663	-0.36	0.7182	r_285(t-2)
	AR2_3_3	0.00704	0.03512	0.20	0.8412	r_436(t-2)
	AR2_3_4	0.06194	0.03907	1.59	0.1132	r_587(t-2)
r_587	CONST4	-0.00069	0.00072	-0.96	0.3370	1
	AR1_4_1	-0.07735	0.03218	-2.40	0.0164	r_216(t-1)
	AR1_4_2	0.02734	0.04237	0.65	0.5189	r_285(t-1)
	AR1_4_3	0.02545	0.03190	0.80	0.4251	r_436(t-1)
	AR1_4_4	0.05548	0.03539	1.57	0.1173	r_587(t-1)
	AR2_4_1	0.05106	0.03214	1.59	0.1125	r_216(t-2)
	AR2_4_2	0.09363	0.04236	2.21	0.0273	r_285(t-2)
	AR2_4_3	-0.02256	0.03190	-0.71	0.4796	r_436(t-2)
	AR2_4_4	-0.04611	0.03549	-1.30	0.1941	r_587(t-2)

误差项协方差矩阵

新息的协方差

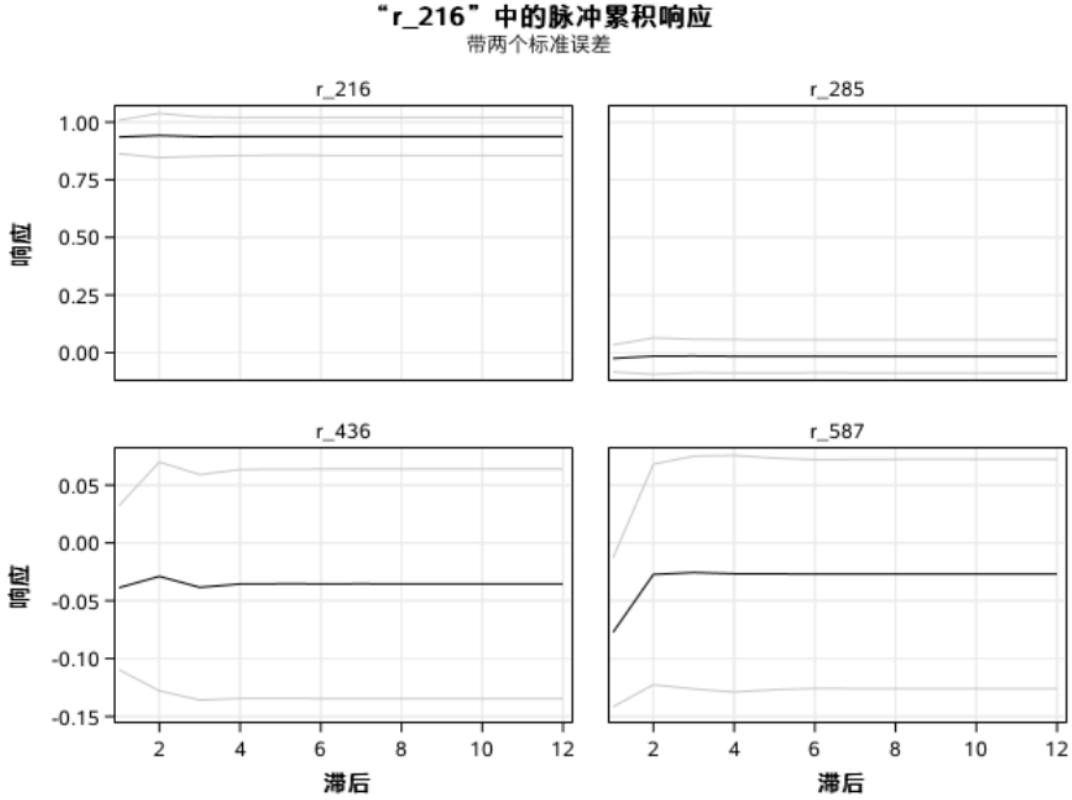
变量	r_216	r_285	r_436	r_587
r_216	0.00069	0.00027	0.00026	0.00023
r_285	0.00027	0.00045	0.00024	0.00023
r_436	0.00026	0.00024	0.00066	0.00020
r_587	0.00023	0.00023	0.00020	0.00055

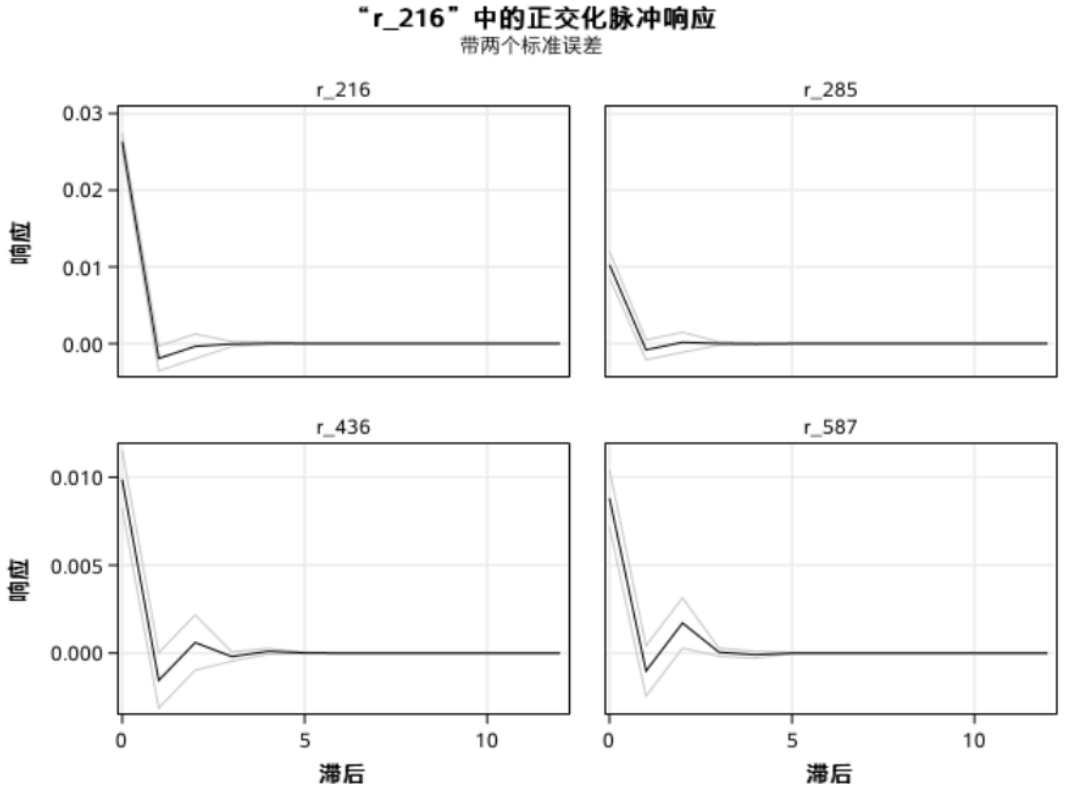
（6）在 proc varmax 语句添加选项 plots=(impulse(accum orth))可以在输出结果中画出累积脉冲响应图和正交脉冲响应图。代码为

```
proc varmax data=c_r plots=(impulse(accum orth));
model r_216 r_285 r_436 r_587/p=2 method=ls ;
run;
```

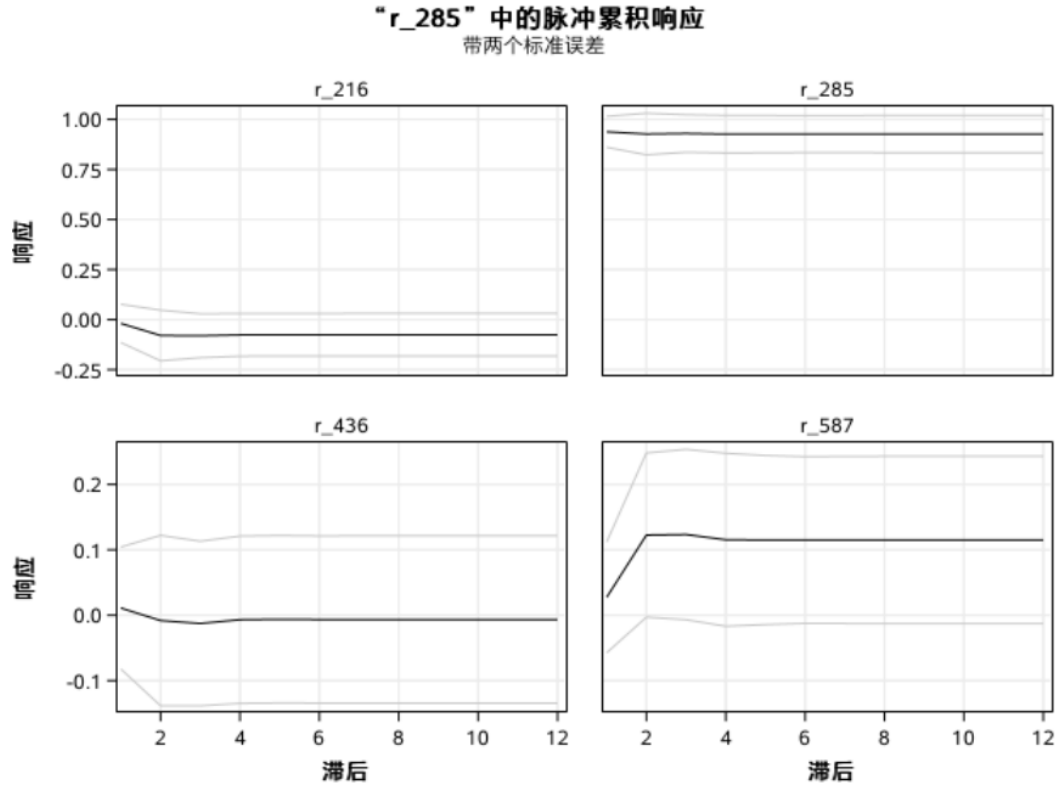
输出结果显示

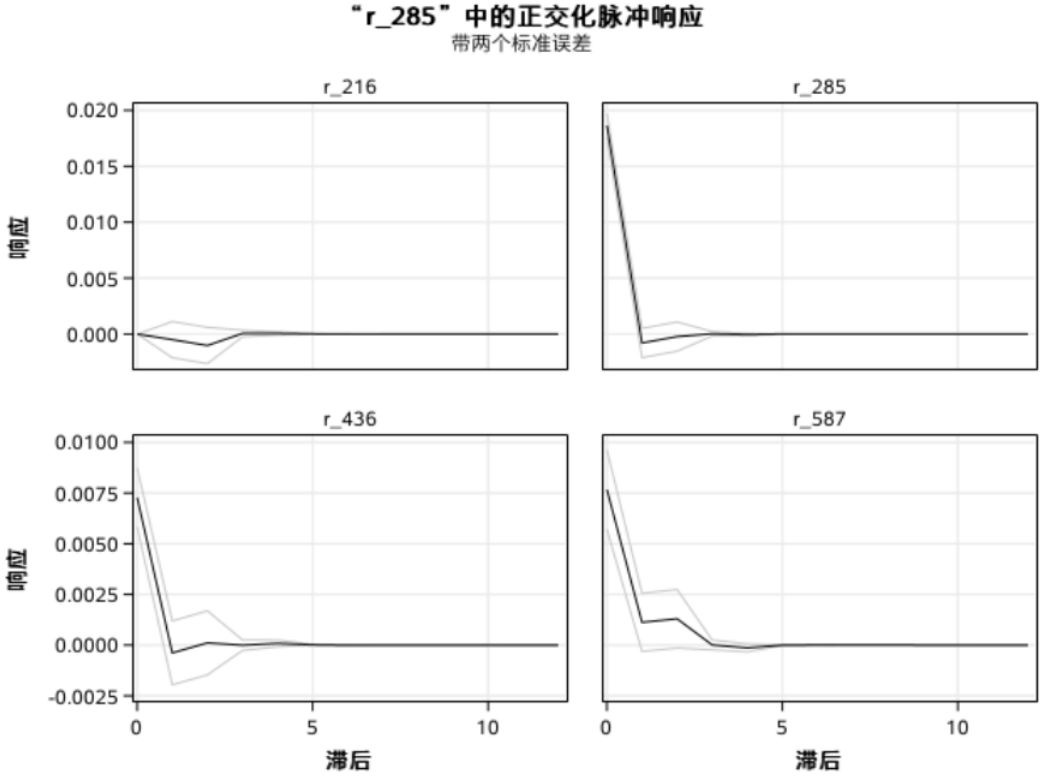
a)r_216 的累积脉冲响应图



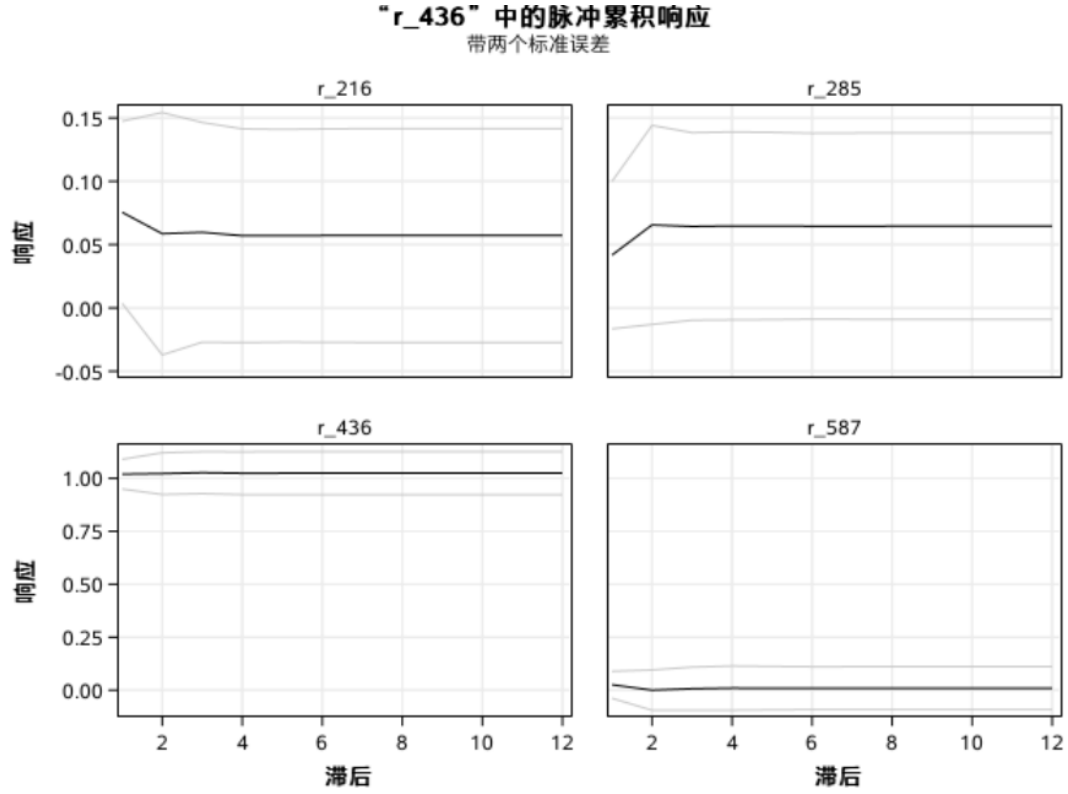


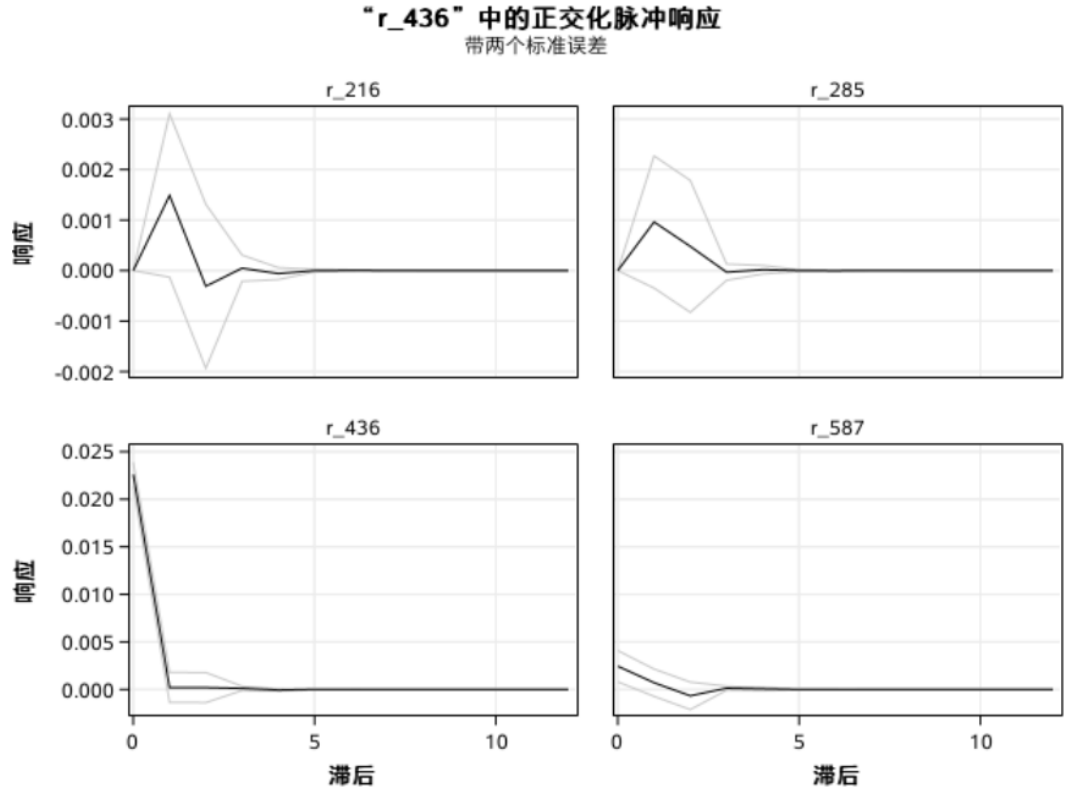
b)r_285 的累积脉冲响应图



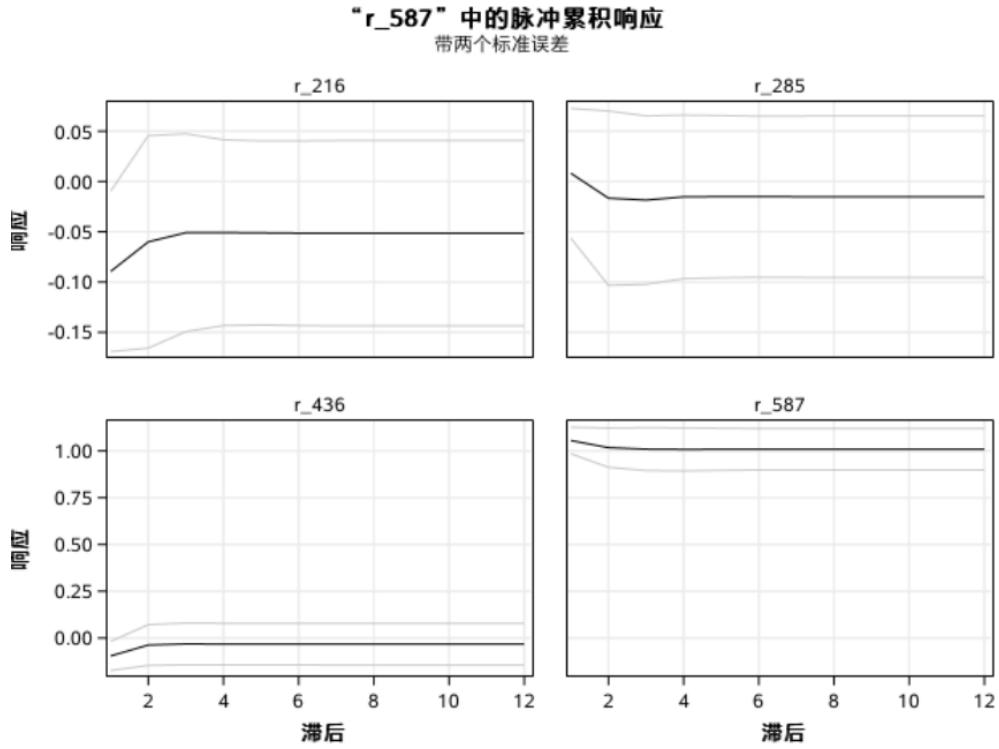


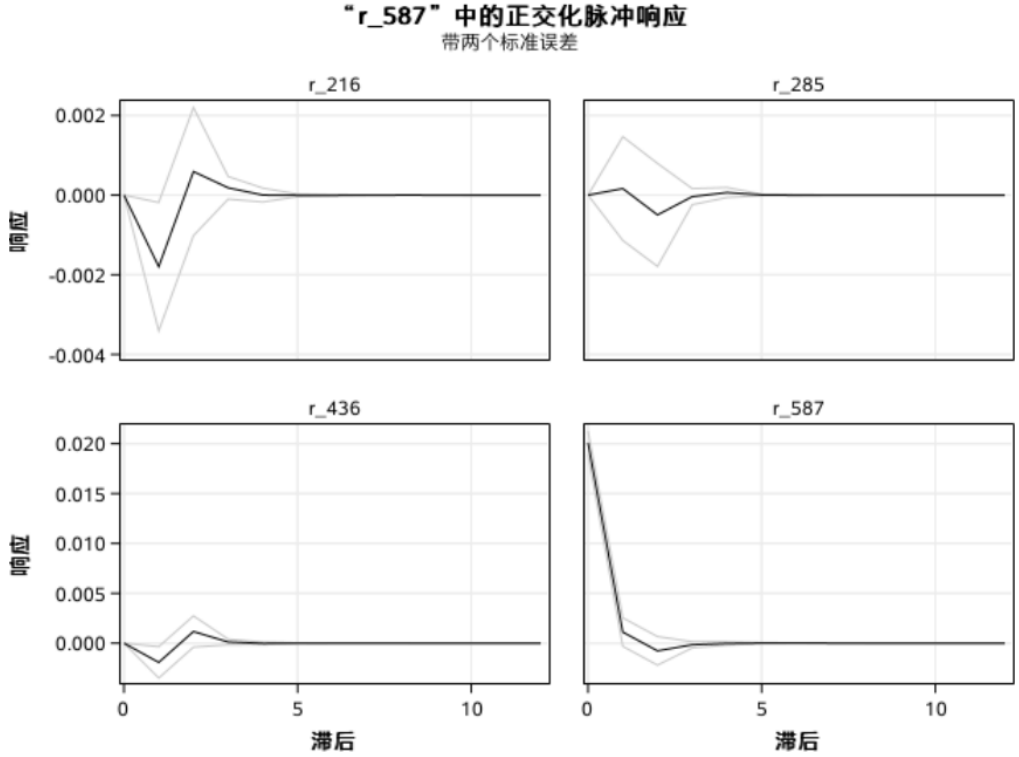
c)r_436 的累积脉冲响应图





（d）r_587 的累积脉冲响应图





（7）在 model 语句添加选项 print=(decompose(5))可以显示 5 期方差分解结果，程序代码为

```
proc varmax data=c_r ;
model r_216 r_285 r_436 r_587/p=2
method=ls print=(decompose(5));
run;
```

输出结果显示为

预测误差协方差的分解

提前	变量	r_216	r_285	r_436	r_587
1	r_216	0.00069	0.00000	0.00000	0.00000
	r_285	0.00011	0.00035	0.00000	0.00000
	r_436	0.00010	0.00005	0.00051	0.00000
	r_587	0.00008	0.00006	0.00001	0.00040
2	r_216	0.00070	0.00000	0.00000	0.00000
	r_285	0.00011	0.00035	0.00000	0.00000
	r_436	0.00010	0.00005	0.00051	0.00000
	r_587	0.00008	0.00006	0.00001	0.00040
3	r_216	0.00070	0.00000	0.00000	0.00000
	r_285	0.00011	0.00035	0.00000	0.00000
	r_436	0.00010	0.00005	0.00051	0.00001
	r_587	0.00008	0.00006	0.00001	0.00041
4	r_216	0.00070	0.00000	0.00000	0.00000
	r_285	0.00011	0.00035	0.00000	0.00000
	r_436	0.00010	0.00005	0.00051	0.00001
	r_587	0.00008	0.00006	0.00001	0.00041
5	r_216	0.00070	0.00000	0.00000	0.00000
	r_285	0.00011	0.00035	0.00000	0.00000
	r_436	0.00010	0.00005	0.00051	0.00001
	r_587	0.00008	0.00006	0.00001	0.00041

还包括

预测误差协方差的比例

提前	变量	r_216	r_285	r_436	r_587
1	r_216	1.00000	0.00000	0.00000	0.00000
	r_285	0.23285	0.76715	0.00000	0.00000
	r_436	0.14710	0.08011	0.77280	0.00000
	r_587	0.14200	0.10789	0.01097	0.73914
2	r_216	0.99191	0.00035	0.00315	0.00459
	r_285	0.23315	0.76476	0.00203	0.00006
	r_436	0.14933	0.07957	0.76555	0.00555
	r_587	0.14284	0.10938	0.01181	0.73597
3	r_216	0.98982	0.00182	0.00328	0.00508
	r_285	0.23293	0.76394	0.00253	0.00060
	r_436	0.14947	0.07938	0.76358	0.00758
	r_587	0.14665	0.11130	0.01249	0.72956
4	r_216	0.98977	0.00183	0.00328	0.00513
	r_285	0.23293	0.76393	0.00253	0.00060
	r_436	0.14951	0.07937	0.76352	0.00760
	r_587	0.14665	0.11129	0.01252	0.72955
5	r_216	0.98976	0.00183	0.00329	0.00513
	r_285	0.23293	0.76392	0.00253	0.00061
	r_436	0.14952	0.07938	0.76350	0.00760
	r_587	0.14665	0.11132	0.01252	0.72950

（8）用 causal 语句可进行变量组之间的 Granger 因果关系检验，代码为

```
proc varmax data=c_r ;
model r_216 r_285 r_436 r_587/p=2 method=ls;
causal group1=(r_216 r_587) group2=(r_285 r_436);
run;
```

检验结果显示为

Granger-Causality Wald 检验

检验	自由度	卡方	Pr > 卡方
1	8	15.33	0.0530

检验 1: 组 1 变量: r_216 r_587

组 2 变量: r_285 r_436

在 5%水平下不能拒绝原假设，即认为浙江医药和新华医疗的日收益不是羚锐制药和片仔癀日收益的 Granger 原因。

4．（1）程序代码为

```
proc sort data=book.shindex out=shindex;
by tdate;
run;
data sh_yy;
merge shindex combine;
by tdate;
if clp & p_216 & p_285 & p_436 & p_587;
run;
```

（2）程序代码为

```
proc arima data=sh_yy;
identify var=clp stationarity=(ADF);
run;
identify var=p_216 stationarity=(ADF);
run;
identify var=p_285 stationarity=(ADF);
run;
identify var=p_436 stationarity=(ADF);
run;
identify var=p_587 stationarity=(ADF);
run;
```

检验结果表明，只有序列 p_587 为平稳序列，结果显示

增广 Dickey-Fuller 单位根检验

类型	滞后	Rho	Pr < Rho	Tau	Pr < Tau	F	Pr > F
零均值	0	-1.6819	0.3704	-2.10	0.0348		
	1	-1.7460	0.3620	-2.14	0.0315		
	2	-1.8226	0.3523	-2.13	0.0322		
单均值	0	-11.9492	0.0841	-3.51	0.0082	7.03	0.0010
	1	-12.7143	0.0698	-3.66	0.0051	7.56	0.0010
	2	-13.8140	0.0532	-3.78	0.0034	7.96	0.0010
趋势	0	-16.9323	0.1275	-3.56	0.0344	7.29	0.0222
	1	-18.1447	0.1005	-3.72	0.0215	7.94	0.0080
	2	-20.0903	0.0679	-3.90	0.0123	8.62	0.0010

其余序列均为单位根过程，为节省篇幅，不再列出结果。

（3）对单位根过程的一阶差分进行单位根检验，程序代码为

```
proc arima data=sh_yy;
identify var=clp(1) stationarity=(ADF);
run;
identify var=p_216(1) stationarity=(ADF);
```

```
run;
identify var=p_285(1) stationarity=(ADF);
run;
identify var=p_436(1) stationarity=(ADF);
run;
```

结果表明，四个一阶差分序列均为平稳序列，表明 clp、p_216、p_285 和 p_436 均为一阶单整序列。将四个变量进行 Johansen 协整检验，

首先选择不包含时间趋势项的选项，程序代码为

```
proc varmax data=book.sh_yy;
model clp p_216 p_285 p_436/p=2 cointest=(johansen);
run;
```

检验结果为

使用轨迹的协整秩检验

H0: Rank=r	H1: Rank>r	特征值	轨迹	Pr > 轨迹	ECM 中的漂移	过程中的漂移
0	0	0.0167	41.9154	0.1590	Constant	Linear
1	1	0.0121	21.5624	0.3220		
2	2	0.0054	6.9299	0.5855		
3	3	0.0003	0.4196	0.5169		

使用限制下的轨迹的协整秩检验

H0: Rank=r	H1: Rank>r	特征值	轨迹	Pr > 轨迹	ECM 中的漂移	过程中的漂移
0	0	0.0199	48.0638	0.1527	Constant	Constant
1	1	0.0121	23.7902	0.4753		
2	2	0.0054	9.1321	0.7235		
3	3	0.0021	2.5786	0.6614		

无论哪种检验模型，得出的结论是不存在协整关系。

以及检验模型是否应该带线性趋势进行检验的检验结果

限制的假设检验

秩	特征值	限定的特征值	自由度	卡方	Pr > 卡方
0	0.0167	0.0199	4	6.15	0.1883
1	0.0121	0.0121	3	2.23	0.5265
2	0.0054	0.0054	2	2.20	0.3325
3	0.0003	0.0021	1	2.16	0.1417

表明应该采用不带线性趋势的检验模型。

再选择包含时间趋势项的选项，程序代码为

```
proc varmax data=book.sh_yy;
model clp p_216 p_285 p_436/p=2
cointest=(johansen) trend=linear;
```

```
run;
```

检验结果表明，不存在协整关系。

对包含 clp 的三个变量一组进行协整分析，发现不存在任何协整关系，对包含 clp 的两个变量一组进行协整分析，发现 clp 和 p_126 存在一个协整关系。

（4）建立 clp 和 p_126 的误差修正模型，代码为

```
proc varmax data=book.sh_yy;
model clp p_436/p=1;
cointeg rank=1;
run;
```

参数估计结果包括：

a）一阶差分模型中的常数项（constnti）和自回归系数矩阵

模型参数估计

方程	参数	估计	标准误差	t 值	Pr > |t|	变量
D_clp	CONST1	28.48110	12.82978	2.22	0.0266	1
	AR1_1_1	-0.00900	0.00406	-2.22	0.0269	clp(t-1)
	AR1_1_2	-0.00913	0.00412	-2.22	0.0269	p_436(t-1)
D_p_436	CONST2	-0.97920	1.04089	-0.94	0.3470	1
	AR1_2_1	0.00037	0.00033	1.11	0.2682	clp(t-1)
	AR1_2_2	0.00037	0.00033	1.11	0.2682	p_436(t-1)

b）误差修正项系数矩阵分解$\Pi = \boldsymbol{\alpha}\boldsymbol{\beta}'$中$\boldsymbol{\alpha},\boldsymbol{\beta}$的估计

Alpha 和 Beta 参数估计

方程	参数	估计	标准误差	t 值	Pr > |t|	变量
D_clp	ALPHA1_1	-2.22499	1.00437	-2.22	0.0269	Beta[,1]'*_DEP_(t-1)
	BETA1_1	0.00405				clp(t-1)
D_p_436	ALPHA2_1	0.09027	0.08149	1.11	0.2682	Beta[,1]'*_DEP_(t-1)
	BETA2_1	0.00410				p_436(t-1)

d）误差项的方差协方差矩阵

新息的协方差

变量	clp	p_436
clp	1217.57605	40.64677
p_436	40.64677	8.01430

（5）检验 p_216、p_285 和 p_436 是否有协整关系，代码为

```
proc varmax data=book.sh_yy;
model p_216 p_285 p_436 /p=1 cointtest=(johansen);
run;
```

检验结果显示有与 i 个协整关系，即

使用轨迹的协整秩检验

H0: Rank=r	H1: Rank>r	特征值	轨迹	Pr > 轨迹	ECM 中的漂移	过程中的漂移
0	0	0.0147	29.1758	0.0585	Constant	Linear
1	1	0.0090	11.3496	0.1906		
2	2	0.0003	0.4209	0.5163		

使用限制下的轨迹的协整秩检验

H0: Rank=r	H1: Rank>r	特征值	轨迹	Pr > 轨迹	ECM 中的漂移	过程中的漂移
0	0	0.0174	35.2819	0.0483	Constant	Constant
1	1	0.0090	14.0972	0.2821		
2	2	0.0026	3.1658	0.5502		

建立三只股票收盘价的误差修正模型，代码为

```
proc varmax data=book.sh_yy;
model p_216 p_285 p_436 /p=1;
cointeg rank=1;
run;
```

参数估计结果为

模型参数估计

方程	参数	估计	标准误差	t 值	Pr > |t|	变量
D_p_216	CONST1	0.14798	0.05684	2.60	0.0093	1
	AR1_1_1	-0.00811	0.00310	-2.61	0.0091	p_216(t-1)
	AR1_1_2	-0.00246	0.00094	-2.61	0.0091	p_285(t-1)
	AR1_1_3	-0.00018	0.00007	-2.61	0.0091	p_436(t-1)
D_p_285	CONST2	0.01945	0.03216	0.60	0.5454	1
	AR1_2_1	-0.00124	0.00176	-0.71	0.4794	p_216(t-1)
	AR1_2_2	-0.00038	0.00053	-0.71	0.4794	p_285(t-1)
	AR1_2_3	-0.00003	0.00004	-0.71	0.4794	p_436(t-1)
D_p_436	CONST3	-0.84529	0.44259	-1.91	0.0564	1
	AR1_3_1	0.05641	0.02416	2.33	0.0197	p_216(t-1)
	AR1_3_2	0.01707	0.00731	2.33	0.0197	p_285(t-1)
	AR1_3_3	0.00122	0.00052	2.33	0.0197	p_436(t-1)

Alpha 和 Beta 参数估计

方程	参数	估计	标准误差	t 值	Pr > |t|	变量
D_p_216	ALPHA1_1	-0.02731	0.01045	-2.61	0.0091	Beta[,1]'*_DEP_(t-1)
	BETA1_1	0.29704				p_216(t-1)
D_p_285	ALPHA2_1	-0.00418	0.00591	-0.71	0.4794	Beta[,1]'*_DEP_(t-1)
	BETA2_1	0.08992				p_285(t-1)
D_p_436	ALPHA3_1	0.18989	0.08134	2.33	0.0197	Beta[,1]'*_DEP_(t-1)
	BETA3_1	0.00642				p_436(t-1)

从中看出，调整系数 alpha2_1 的 t 统计量值为-0.71，概率值为 0.4794，表明该系数不显著。

（6）将 alpha2_1 约束为 0，重新估计误差修正模型，代码为

```
proc varmax data=book.sh_yy;
model p_216 p_285 p_436 /p=1;
cointeg rank=1;
restrict alpha2_1;
run;
```

估计结果为

Alpha 和 Beta 参数估计

方程	参数	估计	标准误差	t 值	Pr > |t|	变量
D_p_216	ALPHA1_1	-0.02723	0.00960	-2.84	0.0046	Beta[,1]'*_DEP_(t-1)
	BETA1_1	0.29763				p_216(t-1)
D_p_285	ALPHA2_1	0.00000	0.00000			Beta[,1]'*_DEP_(t-1)
	BETA2_1	0.09023				p_285(t-1)
D_p_436	ALPHA3_1	0.19097	0.07662	2.49	0.0128	Beta[,1]'*_DEP_(t-1)
	BETA3_1	0.00612				p_436(t-1)

约束条件检验结果为

限定参数的检验

参数	估计	标准误差	t 值	Pr > |t|	方程
Restrict0	-149.54637	167.52996	-0.89	0.3721	ALPHA2_1 = 0

检验结果不能拒绝原假设，表明约束合理。

（7）用 test 语句可以检验约束条件是否成立，此时不需要 restrict 语句，代码为

```
proc varmax data=book.sh_yy;
model p_216 p_285 p_436 /p=1;
cointeg rank=1;
test alpha2_1;
run;
```

检验结果显示

参数的检验

检验	自由度	卡方	Pr > 卡方
1	1	0.50	0.4792

表明不能拒绝原假设。

（8）根据估计结果得出协整系数向量$\boldsymbol{\beta} = (0.2976, 0.09, 0.006)'$，据此得出的协整组合为

$$p_{coint}=0.2976*p\_216+0.09*p\_285+0.006*p\_436$$

用 data 步生成数据集 coint，其中变量 p_coint 为协整组合，代码为

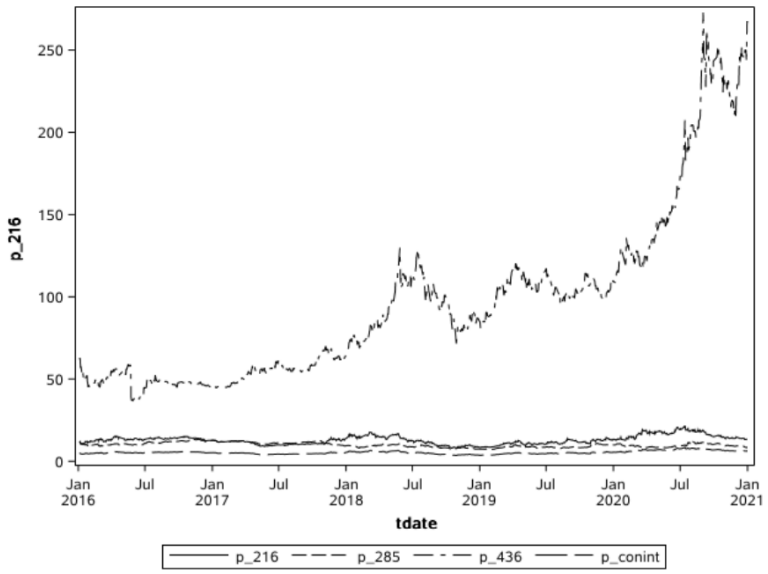
```
data coint;
set book.sh_yy;
p_conint=0.2976*p_216+0.09*p_285+0.006*p_436;
```

```
run;
```

画出三个变量及其协整组合的时序图，代码为

```
proc sgplot data=coint;
series x=tdate y=p_216;
series x=tdate y=p_285;
series x=tdate y=p_436;
series x=tdate y=p_conint;
run;
```

显示为



从中看出，尽管原序列是非平稳的，但协整组合是平稳序列。

5.（1）程序代码为

```
data hts;
set book.sh_security;
rename clp=clp1 trddt=tdate;
if stkcd=600837;
run;
data xhyl;
set book.sh_yiyao;
rename clp=clp2;
if stkcd=600587;
run;
data gsyh;
set book.sh_bank;
rename clp=clp3;
if stkcd=601398;
run;
proc sort data=hts;
by tdate;
proc sort data=xhyl;
by tdate;
```

```
proc sort data=gsyh;
by tdate;
run;
data portf;
merge hts xhyl gsyh;
by tdate;
keep tdate clp1-clp3;
run;
data port_r;
set portf;
r1=dif(log(clp1));
r2=dif(log(clp2));
r3=dif(log(clp3));
if r1 & r2 & r3;
run;
```

（2）首先确定 VAR 阶数，程序代码为

```
proc varmax data=port_r;
model r1-r3/minic=(p=3 q=2 type=HQC);
run;
```

运行结果显示

基于“HQC”的最小信息准则

滞后	MA 0	MA 1	MA 2
AR 0	-24.24541	-24.25482	-24.24983
AR 1	-24.23088	-24.24123	-24.23548
AR 2	-24.20539	-24.23509	-24.21131
AR 3	-24.20211	-24.21115	-24.19185

SAS 系统

VARMAX 过程

模型类型	VMA(1)
估计方法	最大似然估计

建议用一阶向量移动平均模型 VMA(1)。

然后建立 ARMA(0,1)+DCC 模型，代码为

```
proc varmax data=port_r;
model r1-r3/q=1;
garch p=1 q=1 form=dcc subform=EGARCH;
run;
```

参数估计结果分为两部分：

第一部分为 VMA（1）模型中常数项和移动平均项的系数矩阵

模型参数估计

方程	参数	估计	标准误差	t 值	Pr > |t|	变量
r1	CONST1	0.00073	0.00000			1
	MA1_1_1	0.00413	0.03186	0.13	0.8970	
	MA1_1_2	0.08519	0.02528	3.37	0.0008	
	MA1_1_3	-0.00302	0.05565	-0.05	0.9568	
r2	CONST2	-0.00008	0.00000			1
	MA1_2_1	-0.00918	0.03903	-0.24	0.8140	
	MA1_2_2	-0.02811	0.03362	-0.84	0.4033	
	MA1_2_3	0.07103	0.06320	1.12	0.2613	
r3	CONST3	0.00057	0.00000			1
	MA1_3_1	0.02647	0.01925	1.38	0.1693	
	MA1_3_2	0.03364	0.01469	2.29	0.0222	
	MA1_3_3	-0.01486	0.03654	-0.41	0.6844	

第二部分为波动模型参数估计，包括各分量的 EGARCH 模型参数和 DCC 模型中参数

GARCH 模型参数估计

参数	估计	标准误差	t 值	Pr > |t|
DCCA	0.04876	0.00484	10.07	0.0001
DCCB	0.95124	0.00536	177.54	0.0001
DCCS1_2	0.61976	0.00768	80.75	0.0001
DCCS1_3	0.79754	0.00448	177.92	0.0001
DCCS2_3	0.02388	0.01238	1.93	0.0540
GCHC1_1	-5.14192	0.84589	-6.08	0.0001
GCHC2_2	-9.85025	0.00000		
GCHC3_3	-0.81282	0.18110	-4.49	0.0001
ACH1_1_1	-0.00925	0.01077	-0.86	0.3907
ACH1_2_2	0.01869	0.00000		
ACH1_3_3	0.31250	0.04253	7.35	0.0001
EACH1_1_1	-18.67154	21.63941	-0.86	0.3884
EACH1_2_2	-0.07042	0.00000		
EACH1_3_3	0.38960	0.08330	4.68	0.0001
GCH1_1_1	0.31655	0.11130	2.84	0.0045
GCH1_2_2	-0.33767	0.00000		
GCH1_3_3	0.90496	0.02068	43.76	0.0001

（3）从公式（16）可知，proc VARMAX 中多元 GARCH 模型中各分量 EGARCH 模型中表示杠杆效应的参数引用函数名为 EACH(1,1,1)，用 restrict 语句对其进行约束，代码为

```
proc varmax data=port_r;
model r1-r3/q=1;
garch p=1 q=1 form=dcc subform=EGARCH;
restrict each(1,1,1)<0;
run;
```

运行结果显示为

EACH1_1_1	-0.00000	0.00000		
EACH1_2_2	0.04649	0.10330	0.45	0.6528
EACH1_3_3	0.40776	0.08607	4.74	0.0001

（4）test 语句只能检验等式约束，对（3）中的约束条件无法进行检验。提交程序

```
proc varmax data=port_r;
model r1-r3/q=1;
garch p=1 q=1 form=dcc subform=EGARCH;
test each(1,1,1)<0;
run;
```

后信息窗口提示信息为

```
ERROR: Inequalities in the TEST statement are not supported.
The VARMAX procedure stoppedprocessing further step.
```

（5）代码为

```
proc varmax data=port_r;
model r1-r3/q=1;
garch p=1 q=1 form=dcc subform=EGARCH;
restrict ach(1,1,1)+gch(1,1,1)=0.98;
run;
```

参数估计结果中的两个参数满足约束条件

ACH1_1_1	0.11778	0.03399	3.47	0.0006
ACH1_2_2	0.23968	0.04700	5.10	0.0001
ACH1_3_3	0.31790	0.04533	7.01	0.0001
EACH1_1_1	0.59712	0.23011	2.59	0.0096
EACH1_2_2	0.07794	0.10635	0.73	0.4638
EACH1_3_3	0.44514	0.08716	5.11	0.0001
GCH1 1 1	0.86222	0.03399	25.37	0.0001

（6）由于是等式约束，可以进行检验，代码为

```
proc varmax data=port_r;
model r1-r3/q=1;
garch p=1 q=1 form=dcc subform=EGARCH;
test ach(1,1,1)+gch(1,1,1)=0.98;
```

```
run;
```

运行结果显示

参数的检验

检验	自由度	卡方	Pr > 卡方
1	1	34.32	<.0001

检验拒绝原假设。

第九章

1．时变系数 CAPM 模型为

$$r_{z_t} = \alpha_t + \beta_t r_{sh_t} + \epsilon_t, \epsilon_t \sim N(0, \sigma^2)$$

$$\begin{pmatrix}\alpha_t \\ \beta_t\end{pmatrix} = \Phi \begin{pmatrix}\alpha_{t-1} \\ \beta_{t-1}\end{pmatrix} + \begin{pmatrix}\eta_{1t} \\ \eta_{2t}\end{pmatrix}, \begin{pmatrix}\eta_{1t} \\ \eta_{2t}\end{pmatrix} \sim N(\mathbf{0}, Q_t)$$

(1)代码为

```
proc ssm data=book.sh_bys_zsyh_r;
irregular e;
state s(2) cov(d) type=varma(p=1);
v1=1;
component s1=(v1 r_sh)*s;
model r_z=s1 e;
run;
```

参数估计结果为

模型参数估计

成分	类型	参数	估计	标准误差	t 值
s	AR 系数	Phi[1,1]	0.8518720	.	.
s	AR 系数	Phi[1,2]	0.0000406	.	.
s	AR 系数	Phi[2,1]	0.0703492	.	.
s	AR 系数	Phi[2,2]	0.8526384	.	.
s	扰动协方差	Cov[1, 1]	0.0000000	0.0000211	0.00
s	扰动协方差	Cov[2, 2]	0.3748275	.	.
e	不规则	方差	0.0001427	0.0000294	4.86

（2）程序代码为

```
proc ssm data=book.sh_bys_zsyh_r;
irregular e;
state s(2) cov(g) type=varma(p(D)=1);
v1=1;
component s1=(v1 r_sh)*s;
model r_z=s1 e;
run;
```

参数估计结果为

模型参数估计

成分	类型	参数	估计	标准误差	t 值
s	AR 系数	Phi[1,1]	0.221956	0.4259676	0.52
s	AR 系数	Phi[2,2]	0.998492	0.0016454	606.85
s	扰动协方差	RootCov[1, 1]	0.004631	0.0056665	0.82
s	扰动协方差	RootCov[2, 1]	0.029475	0.0346601	0.85
s	扰动协方差	RootCov[2, 2]	0.031434	0.0348676	0.90
e	不规则	方差	0.000146	0.0000515	2.84

(3)程序代码为

```
proc ssm data=book.sh_bys_zsyh_r;
irregular e;
state s(2) cov(d) type=varma(p(D)=1);
v1=1;
component s1=(v1 r_sh)*s;
model r_z=s1 e;
run;
```

参数估计结果

模型参数估计

成分	类型	参数	估计	标准误差	t 值
s	AR 系数	Phi[1,1]	0.1698250	0.4542030	0.37
s	AR 系数	Phi[2,2]	0.9990394	0.0012906	774.08
s	扰动协方差	Cov[1, 1]	0.0000299	0.0000882	0.34
s	扰动协方差	Cov[2, 2]	0.0015132	0.0015181	1.00
e	不规则	方差	0.0001384	0.0000865	1.60

（4）程序代码为

```
proc ssm data=book.sh_bys_zsyh_r;
irregular e;
state s(2) cov(g) type=varma(p=1);
v1=1;
component s1=(v1 r_sh)*s;
model r_z=s1 e;
run;
```

参数估计结果为

模型参数估计

成分	类型	参数	估计	标准误差	t 值
s	AR 系数	Phi[1,1]	0.9459760	.	.
s	AR 系数	Phi[1,2]	0.0000241	.	.
s	AR 系数	Phi[2,1]	0.0413152	.	.
s	AR 系数	Phi[2,2]	0.9428560	.	.
s	扰动协方差	RootCov[1, 1]	0.0000020	3.08E-04	0.01
s	扰动协方差	RootCov[2, 1]	0.0094250	5.80E-02	0.16
s	扰动协方差	RootCov[2, 2]	0.3696670	4.41E-02	8.38
e	不规则	方差	0.0001516	7.08E-06	21.42

（5）四种模型中第三种约束最强，第四种最弱。综合考虑状态变量的可识别性和模型的过度拟合性，第二种模型最为合适。

（6）（2）中模型估计结果中，检测加法离群值的结果输出为

加法离群值汇总

观测	响应变量	估计	标准误差	卡方	Pr > 卡方
354	r_z	0.0669	0.0130	26.61	<.0001
439	r_z	0.0550	0.0130	17.88	<.0001
357	r_z	0.0540	0.0130	17.26	<.0001
462	r_z	-0.0519	0.0131	15.63	<.0001
39	r_z	0.0511	0.0130	15.39	<.0001

为了捕捉离群值的影响，定义 5 各虚拟变量：

```
data test;
set book.sh_bys_zsyh_r;
d1=(_n_=354);d2=(_n_=439);
d3=(_n_=357);d4=(_n_=462);
d5=(_n=_39);
run;
```

将虚拟变量作为回归变量加入观测方程重新估计模型，代码为

```
proc ssm data=test;
irregular e;
state s(2) cov(g) type=varma(p(D)=1);
v1=1;
component s1=(v1 r_sh)*s;
model r_z=d1 d2 d3 d4 d5 s1 e;
run;
```

参数估计结果为

回归参数估计

响应变量	回归变量	估计	标准误差	t 值	Pr > |t|
r_z	d1	0.0672	0.0123	5.44	<.0001
r_z	d2	0.0558	0.0124	4.50	<.0001
r_z	d3	0.0543	0.0124	4.39	<.0001
r_z	d4	-0.0537	0.0125	-4.30	<.0001
r_z	d5	0.0510	0.0124	4.11	<.0001

和

模型参数估计

成分	类型	参数	估计	标准误差	t 值
s	AR 系数	Phi[1,1]	0.0910339	0.332576	0.27
s	AR 系数	Phi[2,2]	0.9973716	0.002309	432.00
s	扰动协方差	RootCov[1, 1]	0.0089417	0.016741	0.53
s	扰动协方差	RootCov[2, 1]	0.0207247	0.036947	0.56
s	扰动协方差	RootCov[2, 2]	0.0597789	0.025286	2.36
e	不规则	方差	0.0000725	0.000296	0.24

（7）程序代码为

```
proc ssm data=test;
irregular e;
state s(2) cov(g) type=varma(p(D)=1) checkbreak;
v1=1;
component s1=(v1 r_sh)*s;
model r_z=d1 d2 d3 d4 d5 s1 e;
run;
```

状态方程结构断点诊断结果为

“s”的元素中断汇总

观测	元素索引	Z 值	Pr > |z|
567	1	3.90	<.0001
1041	1	-3.87	0.0001
334	1	3.84	0.0001
327	1	3.80	0.0001
348	1	-3.71	0.0002

在输入数据集中生成结构变换虚拟变量，为简单期间这里进队第 567 个观测处的结构变化进行处理。

```
data test1;
set test;
```

```
sb=(_n_>=567);
run;
```

在状态方程设定中加入外生变量 sb

```
proc ssm data=test1;
irregular e;
state s(2) cov(g) type=varma(p(D)=1) w(d)=(sb sb);
v1=1;
component s1=(v1 r_sh)*s;
model r_z=d1 d2 d3 d4 d5 s1 e;
run;
```

参数估计结果为

状态方程回归向量的估计

状态	元素索引	估计	标准误差	t 值	Pr > |t|
s	1	0.000403	0.000511	0.79	0.4310
s	2	0.011523	0.003738	3.08	0.0021

表明结构变点发生在第二个（β_t的）方程，第一个（α_t的）方程在对应点发生结构变化不显著。

（8）当状态模型设定为随机游动（rw：random walk），且误差项方差为 0 矩阵，由于误差项期望为 0，等价于误差项为 0，此时的状态模型为

$$\begin{pmatrix}\alpha_t\\ \beta_t\end{pmatrix}=\begin{pmatrix}\alpha_{t-1}\\ \beta_{t-1}\end{pmatrix},t=1,2,\dots,T$$

等价于将状态设定为常数。程序代码为

```
proc ssm data=test1;
irregular e;
state s(2) type=rw;
v1=1;
component s1=(v1 r_sh)*s;
model r_z=d1 d2 d3 d4 d5 s1 e;
run;
```

参数估计结果为

固定状态效应估计

状态	元素索引	估计	标准误差	t 值	Pr > |t|
s	1	0.000612	0.00038	1.61	0.1071
s	2	0.819992	0.03227	25.41	<.0001

轮廓对数似然值（profile loglike）为

似然计算汇总	
统计量	值
使用的非缺失响应值	1159
估计的参数	1
初始化的散射状态元素	7
正规化残差平方和	1152.0013
散射对数似然	3373.682
轮廓对数似然	3400.3072

（2）中模型估计后的轮廓对数似然函数为

似然计算汇总	
统计量	值
使用的非缺失响应值	1159
估计的参数	6
初始化的散射状态元素	0
正规化残差平方和	1159.0003
散射对数似然	3370.2583
轮廓对数似然	3370.2583

将（2）中模型看作无约束模型，（8）中模型看作约束模型，用轮廓对数似然值之差的 2 倍构造似然比检验统计量

$$LR = 2(3400.3072 - 3370.2583) = 30.0498$$

约束条件个数为 5（与（2）相比，（8）中自回归系数矩阵中的约束为 2 个，误差项协方差矩阵约束为 3 个），$LR \sim \chi^2(5)$。查显著水平为 5%的自由度为 5 的χ^2临界值为 11.070，据此可以做出拒绝原假设的结论，即模型（2）更为适合数据。

（9）为了在输出数据集中获得状态向量第二个分量的滤波估计和平滑估计，要用 component 语句将其定义为成分，程序代码为

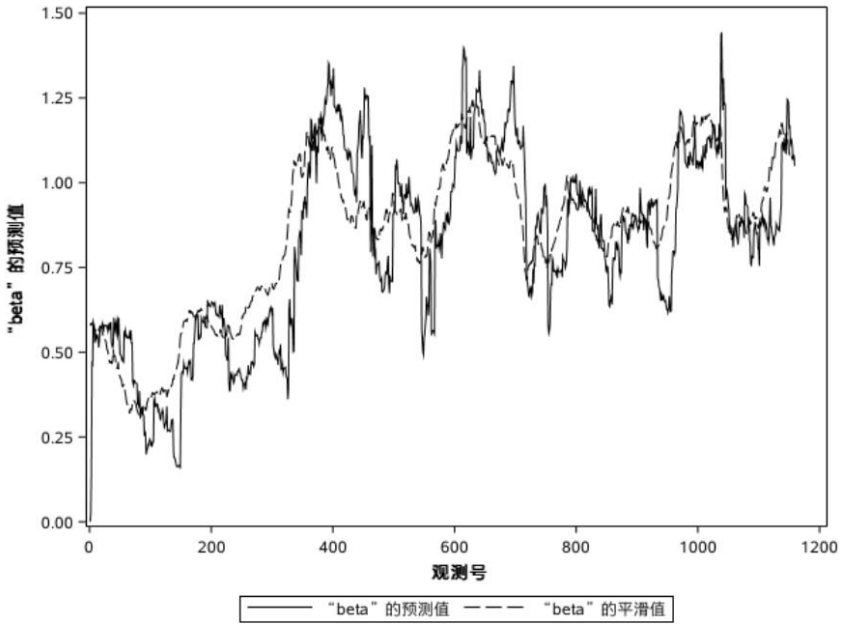
```
proc ssm data=book.sh_bys_zsyh_r;
irregular e;
state s(2) cov(g) type=varma(p(D)=1);
component beta=s[2];
v1=1;
component s1=(v1 r_sh)*s;
model r_z=s1 e;
output out=jg;
run;
```

数据集 jg 中变量 forecast_beta 和 smoothed_beta 分别为β_t滤波序列和平滑序列。用 proc sgplot 画出时序图，代码为

```
proc sgplot data=jg;
series x=obs y=forecast_beta;
```

```
series x=obs y=smoothed_beta;
run;
```

图形输出为



看出两点：1）无论是滤波值还是平滑值，beta 系数都显示出明显的时变特征；2）beta 的平滑值序列比滤波值变化更为平缓。

2．（1）首先用 data 去掉模型变量中包含缺失值的观测

```
data test;
set book.sh trt r;
if r_trt=. or r_m=. then delete;
run;
```

然后用 proc ssm 设定和估计模型

```
proc ssm data=test;
state s(1) type=rw;
comp s1=(r_m)*s[1];
irregular e;
c=1;
model r_trt=c s1 e;
run;
```

state 语句将 s 设定为方差为 0 的随机游动，等价于常数，用 c=1 定义观测方程中常数项对应的变量。参数估计结果为

回归参数估计

响应变量	回归变量	估计	标准误差	t 值	Pr > |t|
r_trt	c	-0.000392	0.000434	-0.90	0.3663

固定状态效应估计

状态	元素索引	估计	标准误差	t 值	Pr > |t|
s	1	0.838	0.0373	22.48	<.0001

模型参数估计

成分	类型	参数	估计	标准误差	t 值
e	不规则	方差	0.000226	9.22E-06	24.54

（2）程序代码为

```
proc autoreg data=test;
model r_trt=r_m;
run;
```

参数估计结果为

参数估计

变量	自由度	估计	标准误差	t 值	近似 Pr > |t|
Intercept	1	-0.000392	0.000434	-0.90	0.3665
r_m	1	0.8383	0.0373	22.48	<.0001

两种方法得出的参数估计十分接近。

3．（1）代码为

```
proc ssm data=book.sh_trt_r;
state s1(1) type=rw;
component comp1=(r_m)*s1[1];
state s2(1) type=varma(p=1) cov(g);
component comp2=s2[1];
c=1;
model r_trt=c comp1 comp2;
run;
```

参数估计结果为

回归参数估计

响应变量	回归变量	估计	标准误差	t 值	Pr > |t|
r_trt	c	-0.000388	0.000414	-0.94	0.3483

固定状态效应估计

状态	元素索引	估计	标准误差	t 值	Pr > |t|
s1	1	0.834	0.0373	22.37	<.0001

模型参数估计

成分	类型	参数	估计	标准误差	t 值
s2	AR 系数	Phi[1,1]	-0.045961	0.0293154	-1.57
s2	扰动协方差	Cov[1, 1]	0.000226	0.0000092	24.54

（2）程序代码为

```
proc autoreg data=book.sh_trt_r;
model r_trt=r_m/nlag=1;
run;
```

参数估计显示为

自回归参数的估计

滞后	系数	标准误差	t 值
1	0.045190	0.028826	1.57

参数估计

变量	自由度	估计	标准误差	t 值	近似 Pr > |t|
Intercept	1	-0.000388	0.000415	-0.94	0.3490
r_m	1	0.8342	0.0373	22.36	<.0001

与（1）结果比较发现，自回归系数估计值 0.04519 和状态空间模型 s2 中 AR 系数 Phi[1,1]=-0.045961 大小十分接近。proc autoreg 采用 model 语句的 nlag 选项设定误差项一阶自回归，自回归系数的符号和标准的回归模型相反。常数项和回归系数的估计值分别为-0.000388 和 0.8342，与状态空间模型估计结果中的 c=-0.000388 和“固定状态效应估计”中的 s1=0.834 相等和接近。

4．程序代码为

```
proc ssm data=cigar;
parms s2g/ lower=(1.e-6);
/*设定区域随机效应变量regioneffect[i], i=1 to 10*/
array RegionArray{10} region1-region10;
do i=1 to 10;
RegionArray[i]=(state=i);
end;
state gamma(10) T(I) Cov1(I)=(s2g);
component regioneffect=gamma*(RegionArray);
/*设定年份效应变量timeEffect*/
state eta(1) type=wn Cov(D);
component timeEffect=eta[1];
intercept=1.0;
irregular wn;
model lsales=intercept lprice lndi
lpimin timeeffect regioneffect wn;
run;
```

参数估计结果显示为

回归参数估计

响应变量	回归变量	估计	标准误差	t 值	Pr > |t|
lsales	intercept	10.399	2.660	3.91	<.0001
lsales	lprice	1.417	0.609	2.33	0.0199
lsales	lndi	-1.016	0.278	-3.66	0.0003
lsales	lpimin	-0.723	0.544	-1.33	0.1838

命名参数的估计

参数	估计	标准误差	t 值
s2g	0.062	0.0402	1.54

模型参数估计

成分	类型	参数	估计	标准误差	t 值
eta	扰动协方差	Cov[1, 1]	0.194	0.0324	5.99
wn	不规则	方差	0.193	.	.

5．（1）将四因子动态 N-S 模型表示为状态空间模型的形式为

$$y_t(\tau) = L_T + \frac{1-e^{-\lambda_1\tau}}{\lambda_1\tau}S_{1,t} + \frac{1-e^{-\lambda_2\tau}}{\lambda_2\tau}S_{2,t} + \left[\frac{1-e^{-\lambda_1\tau}}{\lambda_1\tau} - e^{-\lambda_1\tau}\right]C_t + \epsilon_t(\tau)$$

$$\tau = m_1, m_2, \dots, m_N$$

$$\begin{bmatrix} L_t - \mu_L \\ S_{1,t} - \mu_{S_1} \\ S_{2,t} - \mu_{S_2} \\ C_t - \mu_C \end{bmatrix} = \Phi \begin{bmatrix} L_{t-1} - \mu_L \\ S_{1,t-1} - \mu_{S_1} \\ S_{2,t-1} - \mu_{S_2} \\ C_{t-1} - \mu_C \end{bmatrix} + \begin{bmatrix} \eta_{L,t} \\ \eta_{S_1,t} \\ \eta_{S_2,t} \\ \mu_{C,t} \end{bmatrix}$$

（2）程序代码为

```
proc ssm data=book.yield22 opt(tech=dbldog maxiter=300);
lmd1=0.2;lmd2=2.79;
parms phi1-phi4/lower=-0.99 upper=0.99;
parms sigma1-sigma22/lower=1.e-4;
/* Part I: set 22 error term in measurement eq.*/
array s_array(22) sigma1-sigma22;
do i=1 to 22;
if (type=i) then sigma = s_array[i];
end;
irregular wn variance=sigma;
/* Part II: set factor loadings*/
Z1= 1.0;
```

```
tmp1 = exp(-lmd1*m);
tmp2 = exp(-lmd2*m);
Z2 = (1-tmp1)/(lmd1*m);
Z3 = (1-tmp2)/(lmd2*m);
Z4 = ( 1-tmp1-lmd1*m*tmp1)/(lmd1*m);
/* Part III: set state*/
state gamma(4) T(d)=(phi1 phi2 phi3 phi4) cov(g);
state mu(4) type=rw;
/* Part IV: set measurement eq.*/
comp gammaComp =(Z1-Z4)*gamma;
comp muComp =(Z1-Z4)*mu;
model y= muComp gammaComp wn;
/* Part V: set components for output*/
comp gamma1 = gamma[1];
comp gamma2 = gamma[2];
comp gamma3 = gamma[3];
comp gamma4 = gamma[4];
output out=dnsFor pdv;
run;
```

参数估计结果为

常数项 mu：

固定状态效应估计

状态	元素索引	估计	标准误差	t 值	Pr > |t|
mu	1	4.03	0.0406	99.27	<.0001
mu	2	15.83	1.4970	10.58	<.0001
mu	3	-119.92	10.6360	-11.27	<.0001
mu	4	-16.08	0.8566	-18.78	<.0001

状态方程自回归系数：

命名参数的估计

参数	估计	标准误差	t 值
phi1	0.99000	0.001635	605.42
phi2	0.45369	8.963469	0.05
phi3	-0.03526	.	.
phi4	0.90662	0.025348	35.77

观测方程误差项方差：

sigma1	0.01825	.	.				
sigma2	0.01009	0.013555	0.74				
sigma3	0.00441	0.006057	0.73				
sigma4	0.02154	0.005271	4.09				
sigma5	0.02000	0.004977	4.02	sigma14	0.00359	.	.
sigma6	0.02473	0.004178	5.92	sigma15	0.00805	.	.
sigma7	0.01822	0.003255	5.60	sigma16	0.00010	.	.
sigma8	0.00687	0.006549	1.05	sigma17	0.00010	.	.
sigma9	0.00190	0.004625	0.41	sigma18	0.00010	.	.
sigma10	0.00697	0.002067	3.38	sigma19	0.00010	.	.
sigma11	0.00261	0.000947	2.75	sigma20	0.00010	.	.
sigma12	0.00242	0.001583	1.53	sigma21	0.01610	.	.
sigma13	0.00010	0.003063	0.03	sigma22	0.00823	.	.

状态模型误差项协方差矩阵：

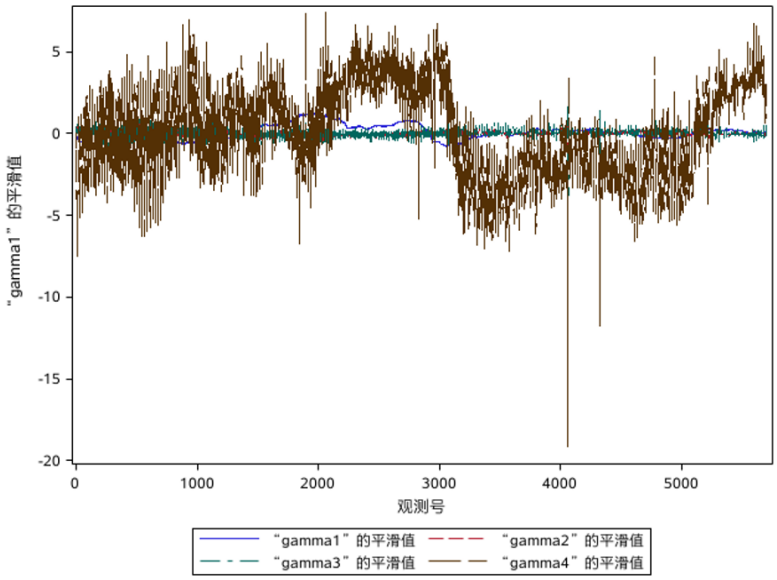
模型参数估计

成分	类型	参数	估计	标准误差	t 值
gamma	扰动协方差	RootCov[1, 1]	0.0336	.	.
gamma	扰动协方差	RootCov[2, 1]	-0.1715	.	.
gamma	扰动协方差	RootCov[2, 2]	0.1545	.	.
gamma	扰动协方差	RootCov[3, 1]	-0.2372	.	.
gamma	扰动协方差	RootCov[3, 2]	0.0150	.	.
gamma	扰动协方差	RootCov[3, 3]	0.9931	.	.
gamma	扰动协方差	RootCov[4, 1]	0.3387	.	.
gamma	扰动协方差	RootCov[4, 2]	-0.7285	.	.
gamma	扰动协方差	RootCov[4, 3]	-0.3087	.	.
gamma	扰动协方差	RootCov[4, 4]	0.9956	.	.

输出数据集中包含了 4 个因子的平滑值，可以用 proc sgplot 画出其时序图，即

```
proc sgplot data=dnsfor;
series x=obs y=smoothed_gamma1;
series x=obs y=smoothed_gamma2;
series x=obs y=smoothed_gamma3;
series x=obs y=smoothed_gamma4;
run;
```

图形显示为



第十章

1.

（1）赋值语句语法正确，但定义的矩阵 b 为字符矩阵，元素为字符'a'和'2'。

（2）出现数据错误，不能用数值矩阵 a 乘以字符矩阵 b。

（3）可以相乘，采用“广播机制”。

（4）相等。按逐元素相乘的“广播机制”进行运算。

（5）不相等，m 是采用矩阵乘法得出的，两个矩阵阶数也不同。

（6）不相等，按逐元素相乘的“广播机制”进行运算理解：d 为行向量，后乘 2×2 矩阵 c 时先将 d 按行广播为 2×2 矩阵，矩阵两行均为向量 d，然后和 c 逐元素相乘。如果时 d 的转置 d`后乘 2×2 矩阵 c，d`时按列广播为 2×2 矩阵后和 c 相乘。

2.

（1）矩阵 a 是行向量。

（2）矩阵 d 定义不正确，字符'A1'和'F3'的前部分不一样。

（3）矩阵 b 中索引算符作用到字符'A'和'F'，可以按字母顺序次序进行，矩阵 e 中的两个字符为汉字，无法顺序进行。

（4）矩阵 c 中两个字符的前部分均为汉字'沈'，可以按后部分的数字字符顺序进行。

3. 选取足够大的正整数 n，用函数 do()对区间(-n,n)进行等分，分割细度可以取为 0.01。例如取 1000。代码为

```
proc iml;
n=1000;
d=0.01;
div=do(-n,n,d);
```

4. 采用矩阵拼接，把矩阵与 2 形成的矩阵垂直拼接。代码为

```
proc iml;
a=2;b=a//{2};
print b;
```

结果显示为

b
2
2

5.

（1）第一种方法：采用比较运算产生矩阵(a>=1)，然后和 a 相乘，代码为

```
proc iml;
a={1 3.2 5,0 0.4 -0.8};
b=(a>=1)#a;
print a b;
```

第二种方法：用函数 loc()发现矩阵(a<1)中非 0 元素位置，然后将 a 中对应位置元素替换为 0，代码为

```
proc iml;
a={1 3.2 5,0 0.4 -0.8};
a[loc(a<1)]=0;
print a ;
```

（2）<:>、>:<分别计算矩阵最大元素的位置和最小元素的位置，<>、><分别计算矩阵最大元素和最小元素。将矩阵 A 的最小值替换为 2 的语句为

```
a[a[>:<]]=2;
```

将 A 的第二列最大元素替换为 4 的语句为

```
a[a[<:>,2],2]=4;
```

（3）用下标降维算符##计算 A 的所有元素平方和的语句为 ss1=A[##]，用逐元素平方然后用函数 sum()求和的语句为 ss2=sum(A##2)。代码为

```
proc iml;
a={1 3.2 5,0 0.4 -0.8};
ss1=a[##];
ss2=sum(a##2);
print ss1 ss2;
```

显示结果为

ss1	ss2
37.04	37.04

两种方法计算结果相同。

6．程序代码为

```
proc iml;
x={0,1,1,1,2,2,3,4,4};
y={1,1,2,6,2,3,3,3,4};
nr=nrow(x);/*compute the number of columns of x*/
X_1=j(nr-1,1,1)||x[1:(nr-1),1];
y_1=y[1:(nr-1),1];
M=inv(X_1`*X_1);
beta=M*X_1`*y_1;
resid=y_1-X_1*beta;/*comput residul*/
print "original fit",beta resid;
```

```
v={1}//x[nr];/*the nineth,i.e.last obervation*/
/*Update the matrix M after removing the 4th obs*/
M=invupdt(M,v,1);
/*get the X and y after removing the 4th obs*/
/*compute new beta and residual*/
X=j(nr,1,1)||x;
beta=M*X`*y;
resid=y-X*beta;
print "After adding Obs. 9", beta resid;
quit;
```

两个 print 语句打印出的结果为

original fit		After adding Obs. 9	
beta	**resid**	**beta**	**resid**
2.1304348	-1.130435	2.0277778	-1.027778
0.2826087	-1.413043	0.375	-1.402778
	-0.413043		-0.402778
	3.5869565		3.5972222
	-0.695652		-0.777778
	0.3043478		0.2222222
	0.0217391		-0.152778
	-0.26087		-0.527778
			0.4722222

可以看出添加第 9 个观测后对回归系数的估计影响和残差的影响。

7.

（1）程序代码为

```
proc iml;
a={1 3.2 5,0 0.4 -0.8,1 5 0.7};
deta=det(a);
call eigen(evalu,evec,a);
sumeig=evalu[+,1];
prod=evalu[#,1];
trace=trace(a);
print deta prod sumeig trace;
```

print 打印结果显示

deta	prod	sumeig	trace
-0.28	-0.28	2.1	2.1

表明矩阵行列式 deta 等于特征根乘积 prod，矩阵迹 trace 等于特征根之后 sumeig。

（2）用函数 inv()，代码为

```
proc iml;
```

```
a={1 3.2 5,0 0.4 -0.8,1 5 0.7};
inverse_a=inv(a);
print inverse_a;
```

结果显示

inverse_a		
-15.28571	-81.28571	16.285714
2.8571429	15.357143	-2.857143
1.4285714	6.4285714	-1.428571

（3）用函数 j()，代码为

```
proc iml;
a={1 3.2 5,0 0.4 -0.8,1 5 0.7};
b=j(nrow(a),ncol(a),0.5);
print b;
```

8.

（1）程序代码为

```
proc iml;
use sashelp.baseball var{name team nhits nhome}
where (name=:'B' & nhits>100 & nhome>15);
read all;
print name;
```

打印结果显示

Name
Bell, Buddy
Brenly, Bob
Buckner, Bill
Baylor, Don
Bell, George
Brett, George
Baines, Harold
Barfield, Jesse
Bass, Kevin
Balboni, Steve
Bream, Sid
Buechele, Steve
Bernazard, Tony
Brunansky, Tom

（2）不能将 name、team、nhits 和 nhome 导入为 IML 形成一个矩阵，因为 name 和 team 为字符变量，其余变量为数值变量，同一矩阵变量的数据类型需相同。能将 name 和 team 导入为 IML 形成一个矩阵，程序代码为

```
proc iml;
```

```
use sashelp.baseball var{name team nhits nhome}
where (name=:'B' & nhits>100 & nhome>15);
read all var{name team} into name_team;
print name_team;
```

打印结果为

name_team	
Bell, Buddy	Cincinnati
Brenly, Bob	San Francisco
Buckner, Bill	Boston
Baylor, Don	Boston
Bell, George	Toronto
Brett, George	Kansas City
Baines, Harold	Chicago
Barfield, Jesse	Toronto
Bass, Kevin	Houston
Balboni, Steve	Kansas City
Bream, Sid	Pittsburgh
Buechele, Steve	Texas
Bernazard, Tony	Cleveland
Brunansky, Tom	Minneapolis

（3）use 语句打开 SAS 数据集供读取，不能写入；Edit 语句打开 SAS 数据集供编辑，即可读取游客写入（修改）。代码为

```
proc iml;
edit sashelp.baseball;
find all where(name=:"P") into obs;
read point obs var{nruns};
nruns=nruns+1;
replace point obs var{nruns};
quit;
```

9．程序代码为

```
proc iml;
a=1.1;
do i=2 to 101;
if i<10 then a=a||(i+i/10);
if (i>10)&(i<100) then a=a||(i+i/100);
if i>=100 then a=a||(i+i/1000);
end;
print a;
create a from a;
```

```
append from a;
quit;
```

10. 在 IML 中对 SAS 数据集进行排序，不需要将数据集读入 IML 的矩阵。如果要在 IML 中使用数据集排序标识 By 组的变量 first.vn 和 last.vn，需要在 data 步内用赋值语句将临时变量 first.vn 和 last.vn 定义为数据集变量，然后将数据集导入 IML 中。

11. 有两个区别。

a) data 步中的 if 语句可以单独使用，对 data 的每次内置循环进行操作选择，并且其作用原理为 if 语句条件成立，则后续语句执行，不成立则 data 短路。IML 中的 if 语句往往需要在 Do-End 循环中使用。

b) IML 中 if 语句中的条件表达式是矩阵表达式，运算结果也是矩阵，表达式运算的结果矩阵中所有元素都是非 0 和非缺失值时，条件才成立。

此外，IML 中没有类似 data 步中 select-when 的多项选择语句。

12. IML 中 Do-End 循环的指标变量初值和终值时矩阵表达式，但表达式运算结果必须是标量（scalar）。此外，IML 没有 Data 步的内置循环。

13. IML 中的符号（symbol）是矩阵名，矩阵一旦定义，其名字存放在符号表（symbol table）中。符号表中的符号只是矩阵的名称，并不是矩阵本身。IML 中需要定义函数和子程序，而每个函数和子程序都会形成自己的局部环境，环境内需要定义矩阵，由此参数局部符号。为了更好地管理不同环境下的符号，实现有序访问，需要设定符号表。符号表包含全局符号表和局部符号表。

14. 程序

```
proc iml;
a=10;b=30;c=40;
start f(x) global(b);
  c=x*b;d=x*a;
  return(c);
finish f;
e=f(a);
```

（1）函数 f() 的局部符号表中并没有矩阵 a，a 是全局符号表中的符号。执行语句 e=f(a) 时，函数内赋值语句 d=x*a; 执行时找不到矩阵 a。

（2）采用 global 语句将全局符号表中的符号 a 能在函数 f() 环境中使用。代码为

```
proc iml;
a=10;b=30;c=40;
start f(x) global(a,b);
c=x*b;d=x*a;
return(c);
finish f;
e=f(c);
print a b c;
```

采用 global(a,b) 子语句后，a 成为函数 f() 内的全局变量，不能再作为自变量。打印结果显示

a	b	c
10	30	40

变量 a、b 和 c 的值没有发生变化。因为 a、b 称为函数 f() 内的全局变量后，f() 内的运算会改变其值，但函数内没有改变 a、b 值的语句。全局变量 c 没有进入 f() 内部，f() 内部的赋值语句 c=x*b 定义的局部变量（符号）c 不影响全局变量 c 的值。print

语句中的 c 是全局变量。

15. 将程序 10-23 中的积分计算定义为函数 integ()，函数自变量包括积分上限 u、下限 l、划分细度 d、最大循环次数 n 和积分精度 tol。代码为

```
proc iml;
start integ(u,l,d,n,tol);
integ1=1;
integ=0;
do i=1 to n until(abs(integ-integ1)<tol);
grid=do(l,u,d);
integ=sum((exp(-(grid##2)/2))#d);
d=0.9*d;
end;
return(integ);
finish;
result=integ(1,0,0.1,100,1e-6);
print result;
```

关键是循环中分割细度 d 的更新，如果按照 d=d/5 的更新规则，则区间分割后形成的向量 grid 的维数很快将达到 10^7 导致数据溢出而无法运行，信息窗口的提示为

```
ERROR: Unable to allocate sufficient memory。
```

16. 程序代码为

```
proc iml;
use book.bteam;
read all var{lname,gender};
read all var{height weight type};
use book.ideal;
read all into ideal;
n=nrow(lname);
obs1=0;obs2=0;
do i=1 to n;
loc=loc(height[i]=ideal[,1]);
stand=ideal[loc,type[i]+1];
lower=stand-5;upper=stand+5;
if (lower<=weight[i])*(weight[i]<=upper) then
do;
obs1=obs1//i;
end;
else
do;
obs2=obs2//i;
end;
end;
obs1=obs1[2:nrow(obs1)];
obs2=obs2[2:nrow(obs2)];
lname_in=lname[obs1];gender_in=gender[obs1];
```

```
height_in=height[obs1];weight_in= weight[obs1];
type_in=type[obs1];
create inshape
var{lname_in gender_in height_in weight_in type_in};
append;
lname_out=lname[obs2];gender_out=gender[obs2];
height_out=height[obs2];weight_out= weight[obs2];
type_out=type[obs2];
create outofshape
var{lname_out gender_out height_out weight_out type_out};
append;
quit;
```

17．自由度为v的 t 分布密度函数为

$$t(\epsilon_t|v)=\frac{\Gamma[(v+1)/2]}{\Gamma(v/2)\sqrt{v\pi}}\left(1+\frac{\epsilon_t^2}{v}\right)^{-(v+1)/2}$$

给定σ_t^2是$e_t=\sigma_t\epsilon_t$的条件分布为

$$t(\epsilon_t|v)=\frac{\Gamma[(v+1)/2]}{\Gamma(v/2)\sqrt{v\pi\sigma_t^2}}\left(1+\frac{e_t^2}{v\sigma_t^2}\right)^{-(v+1)/2}$$

对数似然为

$$\ln t(e_t|v)\propto \ln\Gamma[(v+1)/2]-\ln\Gamma(v/2)$$
$$-\frac{1}{2}\ln\sigma_t^2-\frac{1}{2}\ln v^2-\frac{v+1}{2}\ln\left(1+\frac{e_t^2}{v\sigma_t^2}\right)$$

(1)对数似然为(自由度v为需要估计的参数)

将$e_t^2=(r_trt_t-\beta_0-\beta_1 r_m_t)^2$带入得出

$$\text{ssr(x)}=2T(\ln\Gamma[(v+1)/2]-\ln\Gamma(v/2)-\ln v^2)$$
$$-\sum_{t=1}^{T}\ln(\sigma_t^2)-(v+1)\sum_{t=1}^{T}\ln\left(1+\frac{(r\_trt_t-\beta_0-\beta_1 r\_m_t)^2}{v\sigma_t^2}\right)$$

据此定义函数，代码为

```
t=nrow(r_m);
sigma2=j(t,1,1);
start ssr(x) global(r_trt,r_m,t,sigma2);
    resid2=(r_trt-x[1]#j(t,1,1)-x[2]#r_m)##2;
    sigma2[1]=x[3]/(1-x[4]-x[5]);
    do i=2 to t;
    sigma2[i]=x[3]+x[4]*sigma2[i-1]+x[5]*resid2[i-1];
    end;
    v=x[6];
    temp1=log(gamma((v+1)/2))-log(gamma(v/2))-log(v);
    f1=2*t*temp1;
    f2=sum(log(sigma2<>1e-6));
    f3=(v+1)*sum(log(1+resid2/(v*sigma2)));
```

```
    f=f1-f2-f3;
    return(f);
finish ssr;
```

(2) 为保证方差存在，将自由度约束为$v>2$。约束条件矩阵为

```
cn={. . 0   0    0    2   .  .,
    . . .   0.9 0.9   .   .  .,
    . . .   1    1    .   -1 1};
```

(3) 只需要在 opt 中设定极大化还是极小化目标函数，而这个设定是 opt 向量的第一个分量。因此

```
blc={1};
```

(4) 程序代码为

```
proc iml;
use book.sh_trt_r;
read all var{r_trt,r_m};
t=nrow(r_m);
sigma2=j(t,1,1);
start ssr(x) global(r_trt,r_m,t,sigma2);
    resid2=(r_trt-x[1]#j(t,1,1)-x[2]#r_m)##2;
    sigma2[1]=x[3]/(1-x[4]-x[5]);
    do i=2 to t;
    sigma2[i]=x[3]+x[4]*sigma2[i-1]+x[5]*resid2[i-1];
    end;
    v=x[6];
    temp1=log(gamma((v+1)/2))-log(gamma(v/2))-log(v);
    f1=2*t*temp1;
    sigma2=sigma2<>1e-6;
    f2=sum(log(sigma2));
    f3=(v+1)*sum(log(1+resid2/(v*sigma2)));
    f=f1-f2-f3;
    return(f);
finish ssr;
cn={. . 0   0    0    2   . .,
    . . .   0.9 0.9   .   . .,
    . . .   1    1    .   -1 1};
x0={0 0 0.001 0.5 0.1};
opt={1};
x0={0 0 0.001 0.5 0.1 3};
call nlpnra(rc,xr,"ssr",x0,opt,cn);
print xr;
call nlpdd(rc,xr,"ssr",x0,opt,cn);
print xr;
```

运行结果显示

xr					
-0.001084	0.7862368	4.473E-6	0.7997025	0.0475419	2

xr					
-0.001075	0.7851861	3.572E-6	0.826096	0.0425094	2

两种优化函数得出的记过并不想通过，但差别不大

（5）程序代码为

```
proc iml;
use book.sh_trt_r;
read all var{r_trt,r_m};
t=nrow(r_m);
sigma2=j(t,1,1);
start ssr(x) global(r_trt,r_m,t,sigma2);
    resid2=(r_trt-x[1]#j(t,1,1)-x[2]#r_m)##2;
    sigma2[1]=x[3]/(1-x[4]-x[5]);
    do i=2 to t;
    sigma2[i]=x[3]+x[4]*sigma2[i-1]+x[5]*resid2[i-1];
    end;
    v=x[6];
    temp1=log(gamma((v+1)/2))-log(gamma(v/2))-log(v);
    f1=2*t*temp1;
    sigma2=sigma2<>1e-6;
    f2=sum(log(sigma2));
    f3=(v+1)*sum(log(1+resid2/(v*sigma2)));
    f=f1-f2-f3;
    return(f);
finish ssr;
cn={. . 0   0    0    2   . .,
    . . .   0.9 0.9  .   . .,
    . . .   1    1    .  -1 1};
x0={0 0 0.001 0.5 0.1};
opt={1};
x0={0 0 0.001 0.5 0.1 3};
call nlpdd(rc,xr,"ssr",x0,opt,cn);
call nlpfdd(f,g,hes,"ssr",xr);
cov=inv(-hes);
stde=sqrt(vecdiag(abs(cov)));
t_stat=xr/t(stde);
print xr,t_stat;
```

运行结果显示

xr					
-0.001075	0.7851861	3.572E-6	0.826096	0.0425094	2

t_stat					
-5.751093	38.958277	16.942417	48.150611	5.8557672	27.620265

18．首先用 data 从数据集 sashelp.stocks 中挑选 IBM 股票数据，程序代码为

```
data ibm;
set sashelp.stocks;
if stock='IBM';
run;
```

（1）用子程序 ExportDataSetToR()将数据集 IBM 转换为 R 数据框 IBM，并显示 IBM 内容代码为

```
proc iml;
call ExportDataSetToR("IBM","IBM");
submit/R;
IBM
endsubmit;
```

（2）首先生成序列 p、r1 和 r2，载入 R 程序包 fUnitRoots，然后调用函数 adfTest()对序列 p、r1 和 r2 进行单位根 adf 检验，程序代码为

```
proc iml;
call ExportDataSetToR("IBM","IBM");
submit/R;
p=IBM[,6];
r1=diff(log(p));
r2=diff(p);
require(fUnitRoots)
adfTest(p,lag=4,type=c("c"))
endsubmit;
submit/R;
adfTest(r1);
adfTest(r2);
endsubmit;
```

对 p 的 adf 检验，检验模型滞后阶数为 4，带常数项，但没有时间趋势项。检验结果

```
Title:
 Augmented Dickey-Fuller Test

Test Results:
  PARAMETER:
    Lag Order: 4
  STATISTIC:
    Dickey-Fuller: -2.6353
  P VALUE:
    0.08966

Description:
 Wed Sep 08 16:23:24 2021 by user: SGXMAN
```

p 值 0.0896 不能拒绝原假设，p 存在单位根。

对 r1 和 r2 的 adf 检验，检验模型滞后阶数为 1（取默认值），不带常数项，也没有时间趋势项检验结果分别为 r1：

```
Title:
 Augmented Dickey-Fuller Test

Test Results:
  PARAMETER:
    Lag Order: 1
  STATISTIC:
    Dickey-Fuller: -12.1454
  P VALUE:
    0.01

Description:
 Wed Sep 08 16:23:24 2021 by user: SGXMAN
```

p 值为 0.01，拒绝原假设，表明 r1 为平稳序列。

和 r2

```
Title:
 Augmented Dickey-Fuller Test

Test Results:
  PARAMETER:
    Lag Order: 1
  STATISTIC:
    Dickey-Fuller: -12.5086
  P VALUE:
    0.01

Description:
 Wed Sep 08 17:24:46 2021 by user: SGXMAN
```

p 值为 0.01，拒绝原假设，表明 r1 为平稳序列。

（3）在 R 环境中调用函数 arima()，代码为

```
proc iml;
submit/R;
arima(r2,order=c(4,0,2));
endsubmit;
```

运行结果为

```
Call:
arima(x = r2, order = c(4, 0, 2))

Coefficients:
          ar1     ar2     ar3      ar4      ma1      ma2  intercept
      -0.0703  0.7718  0.2214  -0.0602  -0.1309  -0.8691     0.1360
s.e.   0.0964  0.0820  0.0646   0.0701   0.0742   0.0740     0.1274

sigma^2 estimated as 128.5:  log likelihood = -893.74,  aic = 1803.48
```

（4）和程序 7-4 中第二个 run 组运行结果

无条件最小二乘估计

参数	估计	标准误差	t 值	近似 Pr > |t|	滞后
MU	0.24831	0.36918	0.67	0.5019	0
MA1,1	1.37290	0.38796	3.54	0.0005	1
MA1,2	-0.45365	0.37002	-1.23	0.2215	2
AR1,1	1.14767	0.38565	2.98	0.0032	1
AR1,2	-0.22696	0.29953	-0.76	0.4494	2
AR1,3	0.05391	0.11818	0.46	0.6487	3
AR1,4	-0.14627	0.07105	-2.06	0.0407	4

比较发现：自回归系数和移动平均系数的符号和大小差别较大。

19. （1）-（4）程序代码为

```
proc iml;
call ExportDataSetToR("sashelp.class","class");
submit/R;
age=class[,3];weight=class[,4];height=class[,5];
par(mfcol=c(2,2));
hist(age,plot=TRUE);hist(weight);hist(height);
endsubmit;
```

运行结果显示为