成都理工大学 | 地球物理学院

《地球与空间探测数据处理方法》

第四章 数据的输入和输出

陶 丹

Email：adam.tao@hotmail.com

Address: 北翼楼（地物院）5814室

课程要求和考核方式

课程基础

高等数学/线性代数/概率论与数理统计
高级程序语言与程序设计(C)/Matlab 程序设计

学习目的

1. 掌握IDL可视化分析工具的基础内容，能够独立程序编写并处理地球空间数据。
2. 能够熟悉了解空间探测数据类型及数据文件的存储格式，并能够熟练利用IDL可视化工具来实现空间探测数据的读写、数据格式的转换以及相关可视化分析等。
3. 能够熟练掌握空间探测数据的坐标变化、数据的平滑处理、滤波、谱分析、相关性分析、拟合、平均值统计、中值及误差分析等方法。

考核方式

平时成绩（40%）+ 考试成绩（60%）

其中：

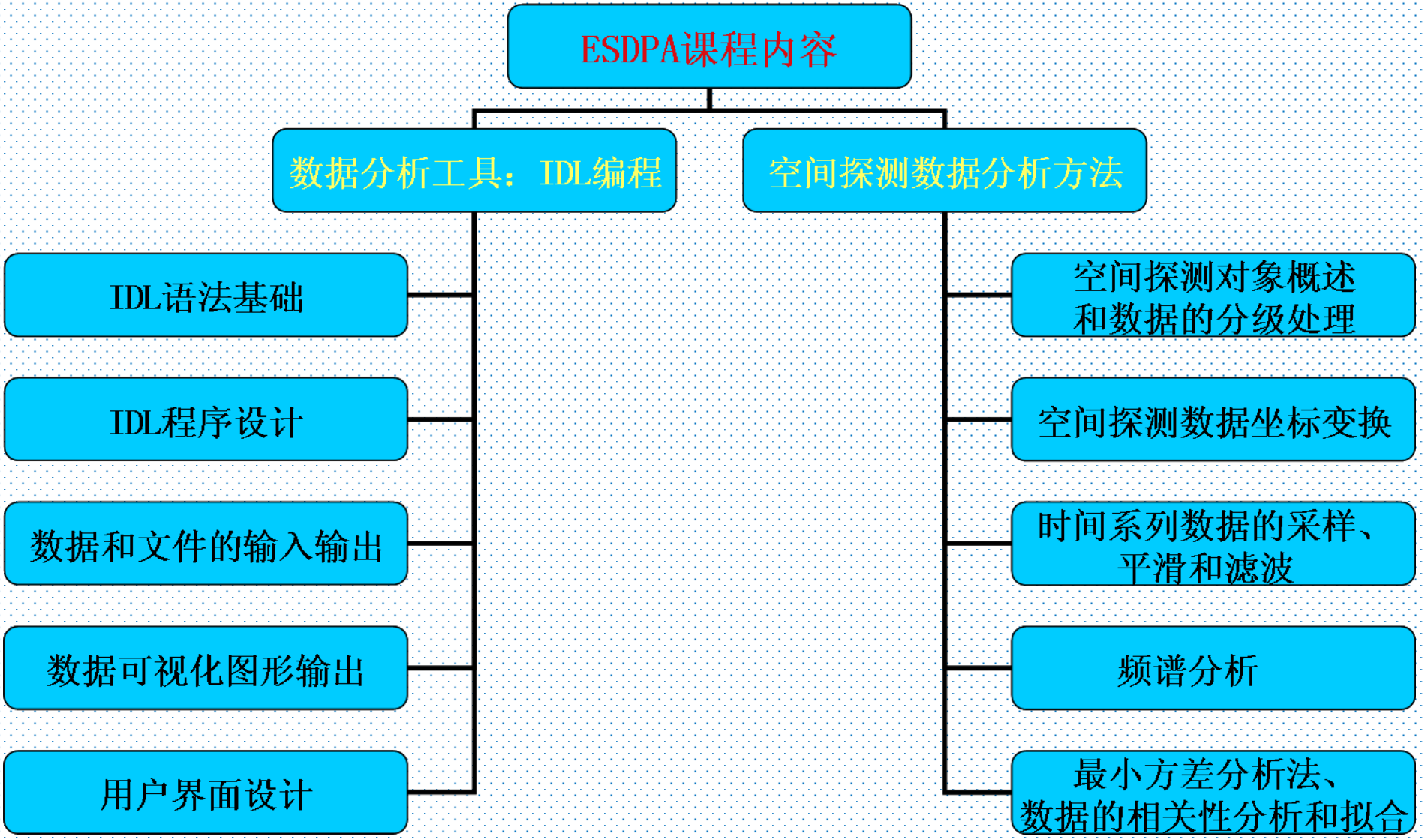
平时成绩（40%）：考勤（10%）+ 大作业/实验报告（30%）

考试成绩（60%）：期末考试（60%，闭卷）

参考书目

- 《IDL可视化工具入门与提高》，闫殿武编著，机械工业出版社，2003.
- 《IDL可视化分析与应用》，韩培友编著，西北工业大学出版社，2006.
- 《IDL程序设计—数据可视化与ENVI二次开发》，董彦卿编著，高等教育出版社，2012.

地球与空间探测数据处理方法课程架构



四、数据的输入和输出：标准输入输出

IDL可以读写的数据格式有：有格式（ASCII）和无格式（二进制），还包括图像的读写等。

4.1 有格式数据的输入输出

- 输入格式：READ, *arg1*, *arg2*,…, *argn*, FORMAT='(格式描述)' [, PROMPT= '提示信息']
- 输出格式：PRINT, *exp1*, *exp2*,…, *expn*, FORMAT='(格式描述)'

例如：

```
IDL> YEAR=0 & MONTH=0 & DAY=0          ; 读入数据前先设定变量类型
IDL> READ, YEAR, MONTH, DAY, PROMPT='请输入年月日：', FORMAT='(I4,1X,I2,1X,I2)'
     请输入年月日：2019 10 31
IDL> PRINT, YEAR, MONTH, DAY, FORMAT='(I4,"/",I2,"/",I2)'
     2019/10/31
```

表 4.1　用于标准 I/O 的过程和函数

名　称	功　　能	名　称	功　　能
print	将有格式数据写到标准输出中	reads	从一个字符串中读取有格式数据
read	从标准输入中读取有格式数据	string( )	将有格式数据写到一个字符串中

四、数据的输入和输出：标准输入输出

- 写入标准输出

PRINT, *exp1, exp2,…, expn*, FORMAT='(格式描述)'

格式代码	格式含义
iN.M	最多N个字符的整数值（.M可选，若有.M，则最右边的M个字符中任何空格位置用0填充）
fN.M	最多N个字符的单精度浮点数，其中小数后M个数字（N=小数点前面的位数+1位小数点+小数点后面的M位数）
dN.M	最多N个字符的双精度浮点数，其中小数点后M位
eN.M	最多N个字符的，指数形式的浮点之，小数点后M个数
gN.M	根据数据大小自动选择科学格式或者f、d格式输出
aN	最多N个字符 的字符串（如果N省略，那么输入字符串中的所有字符将被显示出来
Nx	跳过N个字符的位置
/	换行输出
$	使下一个输出数据输出到该输出数据的所在行的后面（同行输出）
:	如果没有有效的自变量，则终止输出

```
IDL> PRINT,4,FORMAT='(I4.3) '
004
```

```
IDL> PRINT,40000,FORMAT='(G8.2)'
 4.0E+004
```

四、数据的输入和输出：标准输入输出

格式代码	格 式 含 义
[n]A[w]	输入 w 个字符(重复 n 次)若省略 w,输出字符串中所有字符
:	若没有有效的变量,则终止输入输出
$	使下一个输出数据输出到该输出数据所在行的后面(仅用于输出)
[n]I[w]or[n]I[w. m]	输出 w 位十进制整数(其中在最右边的 m 位中的空格位置以 0 填充)
[n]O[w]or[n]O[w. m]	输出 w 位八进制整数(其中在最右边的 m 位中的空格位置以 0 填充)
[n]Z[w]or[n]Z[w. m]	输出 w 位十六进制整数(其中在最右边的 m 位中的空格位置以 0 填充)
[n]F[w. d]	输出 w 位单精度浮点数(其中小数位 d 位)
[n]D[w. d]	输出 w 位双精度浮点数(其中小数位 d 位)
[n]E[w. d]or[n]E[w. dEe]	以指数形式(科学格式)输出 w 位浮点数(其中小数位 d 位,指数位 e 位)
[n]G[w. d]or[n]G[w. dEe]	根据数据大小自动选择科学格式或者 F,D 格式输出
Tn	在当前行的第 n 列(约对位置)输出变量的值
TLn	从当前位置向左移动 n 列,然后输出变量的值
TRn or nX	从当前位置向右移动 n 列,然后输出变量的值
/	换行输出

四、数据的输入和输出：标准输入输出

- 格式代码I、F、D、E或A后面的字符数目N必须包括任何空格、负号、小数点以及指数，而且N只能大于输出数值的字符总数，若小于原数据位数则输出*号表示溢出。

```
IDL> PRINT,200,FORMAT='(I2)'
        **
IDL> PRINT,200,FORMAT='(I5)'
        200
```

- 以同样的格式输出多个值，只要在格式标识符“I、F、D、E、A、X…”前加上输出数据的个数即可。

```
IDL> PRINT,FINDGEN(5),FORMAT='(5F7.3)'
        0.000  1.000  2.000  3.000  4.000
```

- 多个格式数据同时输出，不同格式之间要用“,”隔开，且输出数据之间最好用N个空格隔开。

```
IDL> PRINT,2,!PI,FORMAT='(I4, 2X, F7.3)'
        2    3.142
```

- 可以用括号将多个格式代码重复输出。

```
IDL> PRINT,2,!PI,4,2*!PI,FORMAT='(2(I4,2X,F7.3))'
        2    3.142   4    6.283
```

四、数据的输入和输出：标准输入输出

- 字符串也可以用于格式代码FORMAT( )中，但必须用双引号“”将其括起来。

```
IDL> PRINT,!PI,FORMAT='(" THE VALUE OF PI IS",F9.7)'
      THE VALUE OF PI IS3.1415927
```

- 字符串可以使用格式代码A输出所有字符，或用AN输出N个字符。

```
IDL> PRINT,"DO ONE'S EXERCISE",FORMAT='(A)'
      DO ONE'S EXERCISE
IDL> PRINT,“DO ONE‘S EXERCISE”,FORMAT=’(A7)’      ；格式代码后面的字符数N
      DO ONE'
```

- 如果输出值和设置的格式代码不匹配，则按代码格式输出(假转换)、无输出或输出出错信息。

```
IDL> PRINT,64B,FORMAT='(F5.2)'
      64.00
IDL> PRINT,2,!PI,FORMAT='(A2)'

IDL> PRINT,'HELLO',FORMAT='(I2)'
% TYPE CONVERSION ERROR: UNABLE TO CONVERT GIVEN STRING TO LONG64.
% DETECTED AT: $MAIN$
      0
```

四、数据的输入和输出：标准输入输出

- 多个格式代码同时使用时遵行的法则：
  - 格式代码总是从左到右使用；
  - 如果格式代码数超过需要输出的值的数目，则格式代码从左到右使用，直到所有输出为止；
  - 如果需要输出的值的数目超过格式代码数，则格式代码从左到右使用，当最后一个代码使用完毕之后，代码将再次从头开始使用，但值输出在新的一行中。
    这点不同于输出数值总字符大于格式代码后的总字符数N时输出*表溢出的情况。

```
IDL> PRINT,FINDGEN(6),FORMAT='(2F7.3)'
      0.000  1.000
      2.000  3.000
      4.000  5.000
```

四、数据的输入和输出：标准输入输出

- 从标准输入中读取

READ, *arg1*, *arg2*,…, *argn*, FORMAT='(格式描述)' [, PROMPT='提示信息']

- 为了读取特定类型的数据，要求用户在调用READ前必须先创建该变量的类型。

```
IDL> YEAR=0 & MONTH=0 & DAY=0
IDL> READ, YEAR,MONTH,DAY,PROMPT='请输入年月日：',FORMAT='(I4,1X,I2,1X,I2)'
     请输入年月日：2019 10 31
IDL> PRINT,YEAR,MONTH,DAY,FORMAT='(I4,"/",I2,"/",I2)'
     2019/10/31
```

- 若调用READ前返回的值未定义，而且未使用FORMAT关键字，则返回值默认为浮点型。此时若输入非数字类的值，则出错。

```
IDL> READ,T0,PROMPT='PLEASE ENTER:'
     PLEASE ENTER:89
IDL> HELP,T0
     T0              FLOAT     =       89.0000
```

四、数据的输入和输出：标准输入输出

- 自由格式的输入，即READ语句不使用FORMAT

READ, *arg1*, *arg2*,…, *argn* [, PROMPT=‘提示信息’]

- 输入数字类型（建议使用自由格式输入）
  - 输入值必须使用逗号、空格隔开，或另起一行；
  - 若当前输入行没有包含足够的输入要求的值，则读取下一行，直到读入足够的值为止；
  - 若当前输入行包含了多于输入要求的值，则多余值将被忽略。

```
IDL> A=0
IDL> B=0
IDL> READ,A,B
: 2 3
```

- 输入字符串

每个字符串变量均从独立的行中输入。

```
IDL> A=''
IDL> B=''
IDL> READ,A,B
: SPACE
: PHYSICS
```

四、数据的输入和输出：标准输入输出

从字符串中读取

READS, *arg1, arg2,…, argn*, FORMAT='(格式描述)' [, PROMPT='提示信息']
READ规则同样适用于READS

- **READS可以读取字符串变量，而用STRING()可以写入字符串。**

```
IDL> YEAR=2019
IDL> MONTH=10
IDL> DAY=31
IDL> DATE=STRING(YEAR,MONTH,DAY,FORMAT='(I4,"/",I2,"/",I2.2) ')
IDL> PRINT,DATE
       2019/10/31
IDL> HELP,DATE
       DATE          STRING    = '2019/10/31'

IDL> READS,DATE,YEAR,MONTH,DAY,FORMAT='(I4,1X,I2,1X,I2)'
IDL> PRINT,YEAR,MONTH,DAY
       2019      10      31
```

斜线需要用双括号括起来
格式代码本身就是一个用单引号括起来的字符串

四、数据的输入和输出：标准输入输出

■ 合法的READ自变量

由于READ必须能够调整自变量的大小、类型和值，则支持READ过程的自变量必须以地址传输。
由READ读入的自变量包括标量、变量、数组、结构体、指针及未定义变量等。
而READ调用通过值传递的自变量则是非法的！

```
IDL> ARR0=INTARR(4)
;非法的READ自变量
IDL> READ,ARR0[2],PROMPT='ENTER ARRAY VALUE:'
% READ: EXPRESSION MUST BE NAMED VARIABLE IN THIS CONTEXT: <INT     (     0)>.
% EXECUTION HALTED AT: $MAIN$

;合法的READ自变量
IDL> READ,ARR0,PROMPT='ENTER ARRAY VALUE:'
       ENTER ARRAY VALUE: 0 1 2 3
IDL> PRINT,ARR0
       0      1      2      3
```

四、数据的输入和输出：标准输入输出

- 不同进制数据的转换

计算机可以处理四种进制的数据：二进制、十进制、八进制和十六进制。
分别使用[n]B[w][.m]、[n]I[w][.m]、[n]O[w][.m]和[n]Z[w][.m]进行转换。
格式：FORMAT='([n]X[w][.m])'输出。

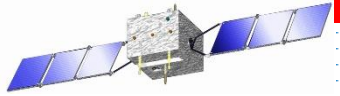
```
IDL> VAR=22
IDL> PRINT,VAR,FORMAT='(B)'
       10110
IDL> PRINT,VAR,FORMAT='(I)'
          22
IDL> PRINT,VAR,FORMAT='(O)'
          26
IDL> PRINT,VAR,FORMAT='(Z)'
          16
IDL> PRINT,12,FORMAT='(Z)'
           C
IDL> PRINT,15,FORMAT='(Z)'
           F
```

4.2 文件操作

- 有格式文件：ASCII文本格式的数据
- 无格式文件：任意类型的二进制数据

表 4.3　处理文件的过程和函数

名　称	功　能	名　称	功　能
openr	打开一个存在的文件，并用于只读	fstat( )	返回一个已打开文件的信息
openw	打开一个用于读写的新文件	eof( )	检测文件末尾
openu	打开一个存在的文件，并用于读写	close	关闭文件
findfile( )	返回当前文件夹中文件的名称	free_lun	释放逻辑设备号，并关闭文件
dialog_pickfile( )	图形文件选择器		



四、数据的输入和输出：打开并读取数据文件操作

- 打开数据文件的命令和格式：

FILE= FILEPATH (‘*FILENAME*’, ROOT_DIR=‘*D:\FOLDER*’) 或:
FILE= DIALOG_PICHFILE (FILTER=‘*.TXT’)

读写的操作是通过逻辑设备号引用文件来进行

OPENR/W/U, LUN, FILE [,/GET_LUN]

- **OPENER:** 打开一个存在的文件并用于只读。
- **OPENW:** 打开一个用于读写的文件，该文件事先存不存在都可以，存在则新文件覆盖旧文件。
- **OPENU:** 打开一个存在的文件并用于读写。

- **LUN:**引用文件的逻辑设备号，它们是-2, -1, 0, … , 128的长整型标量。其中-2, -1, 0分别用于标准错误、标准输出和标准输入；1~99为用户可以使用的逻辑设备号；100~128为使用关键字/GET_LUN时，系统自动分配给文件的逻辑设备号。

通常通过可选关键字/GET_LUN来赋以自由逻辑设备号：在读取数据后用FREE_LUN释放逻辑设备号并关闭文件。其格式为：

```
IDL> LUN=5L
IDL> OPENW,LUN,'TEST.DAT'
IDL> OPENW,LUN,‘TEST.DAT', /GET_LUN
IDL> PRINT,LUN
       100
IDL> FREE_LUN,LUN
```

```
if(n_elements(outfile) eq 1) then begin
    openw,outlun,outfile,/get_lun
endif else begin
    outlun=-1
endelse

printf,outlun,'starting program execution at: ',systime()
```

四、数据的输入和输出：选择和关闭数据文件

- 选择数据文件
- 通过**FILEPATH**命令来设定要打开的数据文件路径**ROOT_DIR**和文件名‘ ’：

  格式：**FILE=FILEPATH(‘*FILENAME*’,ROOT_DIR=‘*D:\FOLDER*’)**

  EXAMPLE-4.1

- 通过**FINDFILE**函数返回当前文件夹中文件的名称：

  • 不调用任何自变量

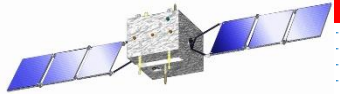
  IDL> PRINT,FINDFILE()

  .\ ..\ example1.pro example2.pro extra_plot.pro FGM_C2_minute.dat get_number.pro hello2.pro hellospace.pro mod_test.pro mypro.sav test.dat test11.pro

  • 使用**STRING**自变量，返回匹配文件名，否则返回空字符串

  IDL> PRINT,FINDFILE('*.DAT')

  FGM_C2_minute.dat test.dat



四、数据的输入和输出：选择和关闭数据文件

- 选择数据文件

- 通过**DIALOG_PICHFILE**函数提供一个标准的打开文件对话框让用户自行查找并选择文件；

  IDL> OPENR, LUN, DIALOG_PICKFILE(),/GET_LUN

  为了缩小查找的文件数量，可以使用关键字***FILTER***来过滤所要查找的数据文件，从而只显示匹配的文件：

  IDL> FILE=DIALOG_PICKFILE(FILTER='*.dat')

  DIALOG_PICKFILE也能用于选择目录名而不仅是一个文件名。设置***DIRECTORY***关键字，在选择窗口内只列出目录而没有文件。

  IDL>DIRECTORY=DIALOG_PICKFILE(/DIRECTORY)

四、数据的输入和输出：选择和关闭数据文件

■ 查看文件的信息

INFO=FSTAT(LUN)

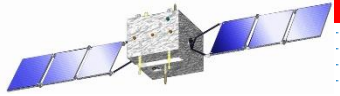
返回文件通道号LUN所引用的文件信息（以结构体返回文件信息）。

EOF(LUN)

检测读取的文件是否到了文件的末尾，是返回‘1’（真），否返回‘0’（假）。

```
IDL> FILE=FILEPATH('FGM_C2_minute.dat',ROOT_DIR='D:\IDLBENCH\IDLLESSON\TEST_DATA')
IDL> OPENR,LUN,FILE,/GET_LUN
IDL> INFO=FSTAT(LUN)
IDL> HELP,INFO.SIZE,INFO.NAME
<Expression>    LONG      =        33
<Expression>    STRING    = 'D:\IDLBENCH\IDLLESSON\TEST_DATA\FGM_C2_minute.DAT'
IDL> PRINT,EOF(LUN)
       0
IDL> DATA=FLTARR(10)
IDL> READU,LUN,DATA
% READU: End of file encountered. Unit: 100
      File: D:\IDLBENCH\IDLLESSON\TEST_DATA\FGM_C2_minute.DAT
% Execution halted at: $MAIN$
IDL> PRINT,EOF(LUN)
       1
IDL> FREE_LUN,LUN
```

POINT_LUN, LUN, 0
将文件指针返回到开头



- 关闭文件且释放逻辑设备号

格式1：**CLOSE, LUN**

关闭通过设备号打开的文件（包括没有使用/GET_LUN获取自由设备号打开的文件），同时释放逻辑设备号。

格式2：**FREE_LUN,LUN**

关闭使用/GET_LUN获取自由设备号打开的文件，同时释放逻辑设备号。

格式3：**CLOSE,/ALL**

关闭所有文件和逻辑设备号。

4.3 读有格式（ASCII）数据文件

格式：**READF, LUN,** ***ARG1,ARG2,…,ARGN*** **[,FORMAT='']**

- 读取大小和长度已知的ASCII码数据文件

```
FILE= FILEPATH ('FILENAME', ROOT_DIR='D:\FOLDER')  ；选择文件
FILE=DIALOG_PICHFILE (FILTER='*.DAT')
OPENR, LUN, FILE, /GET_LUN                          ；打开文件
ARG1=FLTARR(M,N)                                    ；定义一个用于读取数据的变量类型
READF, LUN, ARG1,ARG2,…,ARGN [,FORMAT='']           ；读ASCII文件
FREE_LUN,LUN                                        ；关闭/GET_LUN打开的文件通道号
```

```
IDL>
FILE=FILEPATH('FGM_C2_minute.dat',ROOT_DIR='D:\IDLBENCH\IDLLESSON\TEST_DATA')
IDL> openr,lun,file,/get_lun
IDL> data=fltarr(6)
IDL> readf,lun,data
IDL> print,data
      55.0000      34.0000      67.0000      56.6000      45.0000      56.9000
IDL> free_lun,lun
```

EXAMPLE-4.2

四、数据的输入和输出：读有格式（ASCII）数据文件

- 用**WHILE**循环先检测长度未知的**ASCII**码数据文件中的记录，然后再用**READF**读取整个文件。

例如读取一个不知多长时间的航行位置记录文件***position.dat***：

22:02:13.34UTC -29.9345S -116.0234W 234.2
22:02:14.33UTC -29.9557S -116.0232W 234.1
……

EXAMPLE-4.2　　EXAMPLE-4.3

调用读取文件的函数，并返回一个结构体数组：

```
PRO read_position

input_file=filepath('position.dat',root_dir='D:\IDLBENCH\IDLLESSON\TEST_DATA')
position_data=read_position(input_file)
print,position_data[0].lat,position_data[0].lon
; -29.9345     -116.023

END
```

4.4 写有格式（ASCII）数据文件

完整格式：

OPENW, LUN, FILEPATH (‘*FILENAME*’, ROOT_DIR=‘*E:\FOLDER*’) , /GET_LUN
;建立并打开一个输出文件
***ARG1*=FLTARR(M,N)** ；需要写入的变量赋值
PRINTF, LUN, *ARG1,ARG2,…,ARGN* [,FORMAT=‘’] ；写入数据到LUN通道号打开的文件
FREE_LUN,LUN ；关闭/GET_LUN打开的文件通道号

例如：

```
IDL> DATA=FINDGEN(10,100)
IDL> OPENW,$
LUN,FILEPATH('output.dat',ROOT_DIR='D:\IDLBENCH\IDLLESSON\TEST_DATA'),/GET_LUN
IDL> PRINTF,LUN,DATA,FORMAT='(10F6.2)'
IDL> FREE_LUN,LUN
```

注意事项：在默认情况下，向一个文件中写入有格式的数据只能输出80列，超过将被自动换行写入，从而造成格式上的杂乱。

有效方法：在调用OPENW或OPENU过程中可以通过设置关键字WIDTH=？的数值来输出任意列的数据，这样可以输出固定格式的数据。

EXAMPLE-4.4

■ 读写有格式数据文件

一旦IDL开始为字符串变量读入数据，那么IDL将一直读到行末，且不会停止。

因为在IDL的所有数据类型中，字符串变量可以是任意大小的，且空格符也是ASCII字符。

```
;写入数据到有格式文件
IDL>ARRAY=FINDGEN(25)
IDL>TEXT=['ARRAY', 'VECTOR', 'SCALAR']
IDL>HEADER='TEST DATA FILE.‘
IDL>CREATED_INFO='CREATED BY ADAM @: ' + SYSTIME()
IDL>OPENW , LUN, 'print_and_read_str_from_file.dat',/GET_LUN
IDL>PRINTF, LUN, HEADER                          ;写入头文件
IDL>PRINTF, LUN, CREATED_INFO
IDL>PRINTF, LUN, ARRAY, TEXT                     ;写入数据
IDL>FREE_LUN, LUN

;从有格式文件中读取数据
IDL> HEADER1=STRARR(2)
IDL> DATA1=FLTARR(25)
IDL> TEXT1=''
IDL> OPENR , LUN1, 'PRINT_AND_READ_STR_FROM_FILE.DAT',/GET_LUN
IDL> READF,LUN1,HEADER1,DATA1,TEXT1
IDL> FREE_LUN,LUN1
IDL> PRINT,HEADER1
TEST DATA FILE.‘ CREATED BY ADAM @: Thu Nov 07 17:34:56 2019
```

4.5 读无格式（二进制）数据文件

无格式（二进制）数据文件：

(1) 图像数据文件

(2) 包含多个不同类型数据的二进制文件

■ 读取单一数据类型的无格式文件（图像）：

```
OPENR, LUN, FILEPATH ('FILENAME', ROOT_DIR='E:\FOLDER') , /GET_LUN
;建立并打开一个输出文件
ARG=BYTARR(M,N)                    ；定义一个用于读取数据的变量类型
READU, LUN, ARG                    ；读取LUN通道号文件的数据
FREE_LUN,LUN                       ；关闭/GET_LUN打开的文件通道号
```

例如：

```
IDL> OPENR,LUN,FILEPATH('CTSCAN.DAT',SUBDIR='EXAMPLES/DATA'),/GET_LUN
IDL> DATA=BYTARR(256,256)
IDL> READU,LUN,DATA
IDL> FREE_LUN,LUN
IDL> TVSCL,DATA
```

如果不知道数据的行数，可用FSTAT(LUN)或FILE_LINES查询。

```
IDL> PRINT,FILE_LINES('D:\IDLBENCH\IDLLESSON\MYDEMO\FGM_C2_MINUTE.DAT')
       2
```

4.5 读无格式（二进制）数据文件

无格式（二进制）数据文件：

(1) 图像数据文件

(2) 包含多个不同类型数据的二进制文件

■ 读取单一数据类型的无格式文件（图像）：

如果不知道数据的行数，可用**FSTAT(LUN)**或**FILE_LINES**查询。

```
IDL> PRINT,FILE_LINES('D:\IDLBENCH\IDLLESSON\MYDEMO\FGM_C2_MINUTE.DAT')
       2
```

```
IDL> OPENR,LUN,FILEPATH('CTSCAN.DAT',SUBDIR='EXAMPLES/DATA'),/GET_LUN
IDL> INFO=FSTAT(LUN)
IDL> NCOLS=256L
IDL> NROWS=INFO.SIZE/NCOLS
IDL> DATA=BYTARR(NCOLS,NROWS)
IDL> READU,LUN,DATA
IDL> FREE_LUN,LUN
IDL> TVSCL,DATA
```

四、数据的输入和输出：读写无格式数据文件

- 读取混合数据类型的无格式文件（二进制数据）：
  先用**WHILE**检测文件的长度，再用“**READU, LUN, 匿名结构体**”读取数据。

EXAMPLE-4.5

- 将变量存储为一个无格式的数据文件（创建一个无格式文件）

格式：**WRITEU,LUN,*ARG1,ARG2,…***

```
IDL> DATA=DIST(256)
IDL> INFO=SIZE(DATA)
IDL> INFO_EXTRA='FILE WAS CREATED BY ADAM AT:'
IDL> FILE=FILEPATH('writeufile.dat',ROOT_DIR='D:\IDLBENCH\IDLLESSON\TEST_DATA')
IDL> OPENW,LUN,FILE,/GET_LUN
IDL> WRITEU,LUN,INFO_EXTRA,SYSTIME()
IDL> WRITEU,LUN,INFO
IDL> WRITEU,LUN,DATA
IDL> CLOSE,/ALL
```

四、数据的输入和输出：读写无格式数据文件

- 多个不同类型数据同时写入一个无格式文件时按先后顺序进行。

例如写入头文件和数据到同一个无格式数据文件中：

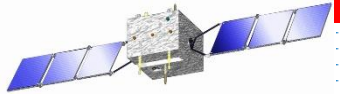
```
IDL> DATA=DIST(256)
IDL> INFO=SIZE(DATA)
IDL> PRINT,INFO
       2       256       256         4      65536
IDL> FILE=FILEPATH('writeufile.dat',ROOT_DIR='D:\IDLBENCH\IDLLESSON\TEST_DATA')
IDL> OPENW,LUN,FILE,GET_LUN
IDL> DATA=DIST(256)
IDL> INFO=SIZE(DATA)
IDL> PRINT,INFO
       2       256       256         4      65536
IDL> INFO_SIZE=N_ELEMENTS(INFO)
IDL> PRINT,INFO_SIZE
       5
IDL> HEADER=[INFO_SIZE,INFO]
IDL> PRINT,HEADER
       5       2       256       256         4      65536
IDL> OPENW,LUN,DIALOG_PICKFILE(),/GET_LUN
IDL> WRITEU,LUN,HEADER        ；先写入头文件
IDL> WRITEU,LUN,DATA          ；后写入数据
IDL> FREE_LUN,LUN
```

四、数据的输入和输出：读写无格式数据文件

- 从同一个无格式数据文件中读取不同类型的数据时也按先后顺序进行（知道数据维度和大小）。

例如从同一个无格式数据文件分别读取文件头和数据：

```
IDL>
FILE=FILEPATH('writeufile.dat',ROOT_DIR='D:\IDLBENCH\IDLLESSON\TEST_DATA')
IDL> OPENR,LUN,FILE,GET_LUN
IDL> HEADER=LONARR(6)
IDL> READU,LUN,HEADER                    ；先读文件头
IDL> PRINT,HEADER
       5        2      256      256        4    65536
IDL> DATA=LONARR(256,256)
IDL> READU,LUN,DATA                      ；后读数据
IDL> FREE_LUN,LUN
```



四、数据的输入和输出：读CDF数据文件

- 首先通过CDF文件信息查询软件（CDFToolsDriver）浏览数据文件的物理量名称，然后调用LOADCDF过程直接读取变量。

格式：LOADCDF, *FILENAME, 'VARIABLE NAME IN THE CDF FILE', VARIABLE_DEFINED_BY_USER*

```
FILENAME='E:\FOLDER\*.CDF'
LOADCDF, FILENAME, 'THG_ASK_ATHA_EPOCH', TIME
LOADCDF, FILENAME, 'THG_ASK_PGEO', PGEO  ;COUNTS
LOADCDF, FILENAME, 'THG_ASK_ROW', PIXEL
```

EPOCH时间转换格式：

```
CDF_EPOCH,TIME[I],YR,MO,DY,HR,MN,SC,MILLI,/BREAKDOWN
```

四、数据的输入和输出：读CDF数据文件

- 首先通过CDF文件信息查询软件（**CDFToolsDriver**）浏览数据文件的物理量名称，然后调用**LOADCDF**过程直接读取变量。

格式：**LOADCDF**, *FILENAME*, '*VARIABLE NAME IN THE CDF FILE*',
VARIABLE_DEFINED_BY_USER

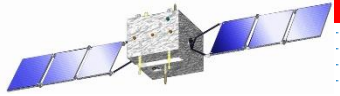
```
PRO read_cdf_file

loadcdf, 'D:\IDLBENCH\IDLLESSON\TEST_DATA\C1_CP_CIS-
HIA_HS_1D_PEF__20020413_190000_20020414_210000_V090718.cdf','time_tags__C1_CP_
CIS-HIA_HS_1D_PEF',TIME

nrecords=n_elements(time)
;print,nrecords
hour=fltarr(nrecords)
second=fltarr(nrecords)

for i=0L,nrecords-1L DO BEGIN
   CDF_EPOCH, time[i], yr, mo, dy, hr, mn, sc, milli, /breakdown
   hour[i]=hr+mn/60.0+sc/3600.0+milli/3.6e6
   second[i]=sc+milli/1.0e3
   print,time[i],hour[i],second[i]
endfor

END
```



四、数据的输入和输出：读取和展示图片

IDL支持的图像文件类型包括：BMP（位图）、DICOM（医学数字图像）、GEOTIFF（包含地理数据的TIFF文件）、JPEG、MPEG、PNG等。

- 读取函数：READ_BMP (*FILENAME*)、READ_PPM (*FILENAME*) 、READ_DICOM (*FILENAME*) 、READ_PICT (*FILENAME*) 、READ_PNG (*FILENAME*) 、 READ_TIFF (*FILENAME*) 、READ_JPEG (*FILENAME*) 、READ_BINARY (*FILENAME*)

- 通用的图片读取命令：READ_IMAGE(*FILE*) ；该命令打开的图像件用TV显示时不变色。

例如：

```
IDL> FILE = DIALOG_PICKFILE(FILTER='*.JPG', /MUST_EXIST)
IDL> IMAGE=READ_IMAGE(FILE)
IDL> TV,IMAGE,TRUE=1      ；在窗口展示所读取的图片
```

亦可以用DIALOG_READ_IMAGE函数读取JPG文件。

使用格式为：

```
YN
=DIALOG_READ_IMAGE(FILENAME,FILTER_TYPE=STRING,IMAGE=VARIABLE,TITLE=STRING )
```

例如：*YN = DIALOG_READ_IMAGE(FILTER_TYPE= 'JPG', IMAGE=IDATA)*

四、数据的输入和输出：按指定格式保存图像文件

- 保存图片的函数有：**WRITE_BMP *(filename)*** 、**WRITE_PICT *(filename)*** 、**WRITE_PNG *(filename)*** 、**WRITE_TIFF *(filename)*** 、**WRITE_JPEG *(filename)*** 、**WRITE_PPM *(filename)***等

通用的图片保存命令：**WRITE_IMAGE, *filename*, FORMAT, *data***

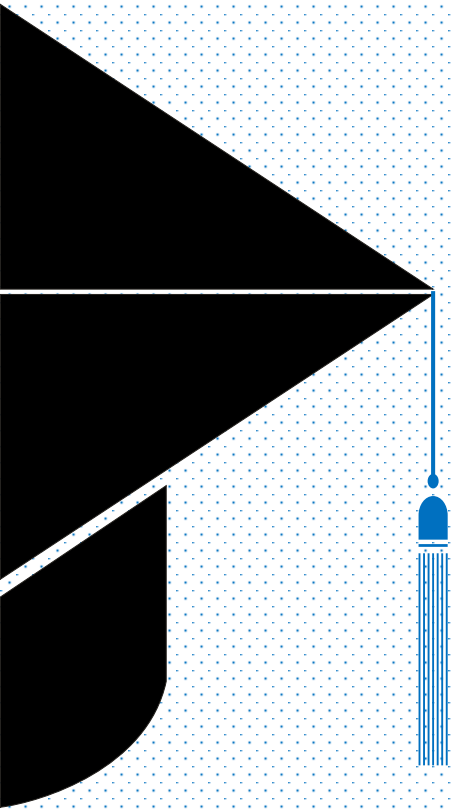
也可以通过**DIALOG_WRITE_IMAGE**对话框存储图像文件，使用格式为：

YN = DIALOG_WRITE_IMAGE(IMAGE, FILE=STRING, TITLE=STRING , TYPE=VARIABLE)

例如：

```
IDL> FILE = DIALOG_PICKFILE(FILTER= '*.JPEG', /MUST_EXIST)
IDL> IMAGE=READ_IMAGE(FILE)
IDL> TV,IMAGE,TRUE=1
IDL> WRITE_PNG, 'WRITE_PNG.PNG', IMAGE
IDL> WRITE_IMAGE, 'WRITE_BMP.BMP', 'BMP',IMAGE
IDL> RESULT = DIALOG_WRITE_IMAGE(IMAGE, TYPE=".BMP")
```

成都理工大學
CHENGDU UNIVERSITY OF TECHNOLOGY



演示完毕

Thank You