数学软件 Matlab

—— 文件操作

内容提要

- 文件的打开与关闭
- 文本文件的写与读
- 二进制文件的写与读

Matlab 文件操作介绍

- 文件操作是一种重要的输入输出方式，Matlab 提供了一系列输入输出函数，专门用于文件操作。

- Matlab中的输入输出函数是以 C 语言标准库函数中的输入输出函数为基础开发的，所以这些函数与 C 语言的输入输出函数相类似。

- Matlab文件操作三步骤：

  (1) 打开文件
  (2) 对文件进行读写操作
  (3) 关闭文件

文件的打开

- 文件的打开

```
fid=fopen(文件名，打开方式)
```

  - 其中文件名用字符串形式表示（可以带路径名）
  - 打开方式有（若不指定打开方式，则表示只读）：

'r'	只读，文件必须存在（缺省的打开方式）
'w'	写文件，若文件已存在则原内容将被覆盖；若文件不存在则新建一个
'a'	在文件末尾添加，文件若不存在则新建一个
'r+'	可读可写，文件必须存在
'w+'	可读可写，若文件已存在则原内容将被覆盖；若文件不存在则新建一个
'a+'	可读可写可添加，文件若不存在则新建一个

文件的打开

```
fid=fopen(文件名，打开方式)
```

- fid 为文件句柄，通过它才能对该文件进行操作
- 如果句柄值大于 0，则表示文件打开成功；
- 若文件打开失败，fid 的返回值为 -1。

```
fid=fopen('output.txt', 'wt+');
fprintf(fid,'Hello world!\n');
fclose(fid);
```

- 有两个标准代码文件，不需打开就可以直接使用，分别为：

fid=1 标准输出文件，fid=2 标准错误文件。

```
fprintf(1,'Hello world!\n');
fprintf(2,'Hello world!\n');
```

文件的关闭

- 文件的关闭

```
status=fclose(fid);
```

  - 其中 `fid` 为所要关闭的文件的句柄
  - `status` 为关闭文件的返回代码，若成功则为 `0`，否则为 `-1`
  - 文件操作结束后一定要关闭

内容提要

- 文件的打开与关闭
- 文本文件的写与读
- 二进制文件的写与读

文本文件的写入

- 向文本文件中写数据

```
count=fprintf(fid,format,输出变量列表)
```

  - 将指定的变量按指定的格式写入文本文件中
  - 若省略 fid，则表示在屏幕上输出
  - count 返回所写入的数据的个数（可以省略）
  - format 用来指定写数据时采用的格式，通常由三部分组成

    普通字符串、格式字符串、转义字符

```
x=3.14; str='math';
fid=fopen('output.txt','wt');
fprintf(fid,'x=%f, str=%s\n', x, str);
fclose(fid);
```

文本文件的写入

- 数据输出时采用的格式（format）

(1) 普通字符串：按原样输出

(2) 格式字符串：指定变量的输出格式，以 % 开头，包括

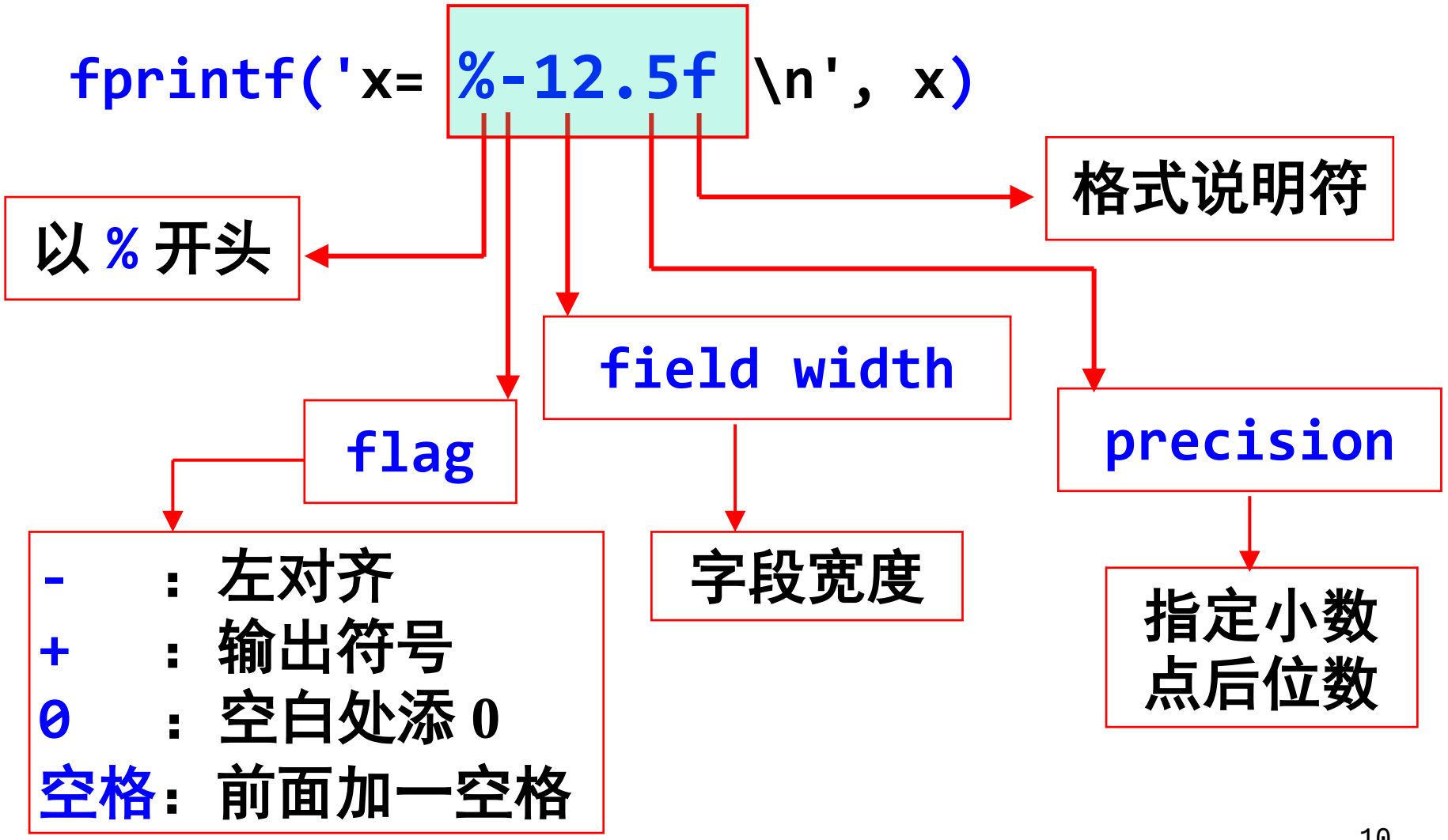
- flags（可选）
- width and precision（可选）
- conversion character（格式说明符，必须）

(3) 转义字符：输出特殊符号，如换行等，常见的有：

\b	退后一格	\t	水平制表符
\f	换页	\\	反斜杠
\n	换行	''	单引号
\r	回车	%%	百分号

格式字符串

- 格式字符串示例



格式说明符

- 常见的格式说明符

c	字符型	g	浮点数（自动）
d	十进制整数	o	八进制
e	浮点数（科学计数法）	s	字符串
f	浮点数（小数形式）	x/X	十六进制

例：

```
x=sqrt(2);
fid=fopen('out2.txt','wt');
fprintf(fid,'x=%12.5f, \n', x);
fprintf(fid,'x=%-12.5f, \n', x);
fclose(fid);
```

文本文件写入举例

例：

```
x=0:0.1:1;  y=exp(x);
fid=fopen('output.txt','wt');
fprintf(fid,'   x       exp(x)\n');
for k=1:length(x)
  fprintf(fid,'%6.2f %12.8f\n', x(k),y(k));
end
fclose(fid);
```

- **注：格式的重复使用**

```
x=0:0.1:1;  y=exp(x); z=[x; y];
fid=fopen('output.txt','wt');
fprintf(fid,'   x       exp(x)\n');
fprintf(fid,'%6.2f %12.8f\n', z);
fclose(fid);
```

文本文件的读取

- 从文本文件中读取数据

```
[A,count]=fscanf(fid,format,size)
        A=fscanf(fid,format,size)
```

  - A 用来存放读取的数据
  - count 返回读取数据的个数，为可选项
  - fid 为文件句柄
  - size 为可选项，若缺省，则读取整个文件，若给出，则取值可以是：

N	读取 N 个数据，组成一个列向量
Inf	读取整个文件，组成一个列向量
[m,n]	读取 m × n 个数据，组成到一个 m × n 矩阵，按列存放

文本文件读取举例

例：

```
x=0:0.1:1;
y=exp(x);
z=[x; y];

% 写文件
fid=fopen('output.txt','wt');
fprintf(fid,'%6.2f %12.8f\n', z);
status=fclose(fid);

% 读文件
fid=fopen('output.txt','rt');
A=fscanf(fid,'%f');  % 注意格式字符串
% A=fscanf(fid,'%f',[2,11]);
status=fclose(fid);
```

内容提要

- 文件的打开与关闭
- 文本文件的写与读
- 二进制文件的写与读

二进制文件的写入

- 向二进制文件中写入数据

```
count=fwrite(fid,A,precision)
```

  - 按指定的数据类型将矩阵 A 中的元素写入到文件中。其中 count 返回所写入的数据元素个数（可省略）

例：

```
A=magic(5);
fid=fopen('magic5.dat','wb');
fwrite(fid,A,'int8');
fclose(fid);

fid=fopen('magic5.dat','rb');
[B,count]=fread(fid,[5,inf],'int8');
fclose(fid);
```

二进制文件

- **precision 代表写入的数据的类型，缺省为 uchar**

'uchar'	无符号字符	'uint16'	16位无符号整数
'schar'	带符号字符	'uint32'	32位无符号整数
'int8'	8位带符号整数	'uint64'	64位无符号整数
'int16'	16位带符号整数	'float32'	32位浮点数
'int32'	32位带符号整数	'float64'	64位浮点数
'int64'	64位带符号整数	'double'	64位双精度数
'uint8'	8位无符号整数		

二进制文件

- 以下数据类型与使用的平台有关：

'char'	带符号字符
'short'	16位带符号整数
'int'	32位带符号整数
'long'	32或64位带符号整数
'ushort'	16位无符号整数
'uint'	32位无符号整数
'ulong'	32或64位无符号整数
'float'	32位浮点数

二进制文件的读

- 从二进制文件中读取数据

```
[A,count]=fread(fid,size,precision)
        A=fread(fid,size,precision)
```

- A 用来存放读取的数据
- count 返回读取数据的个数，可选项
- fid 为文件句柄
- size 为可选项，缺省为读取整个文件。取值可以是：

N	读取 N 个数据，组成一个列向量
Inf	读取整个文件，组成一个列向量
[m,n]	读取 m × n 个数据到一个 m × n 矩阵中，按列存放

二进制文件读写举例

例：

```
fid=fopen('output.dat','rb');
A=fread(fid,100,'double');
status=fclose(fid);
```

```
fid=fopen('output.dat','rb');
[A,count]=fread(fid,[100,100],'double');
status=fclose(fid);
```

读写的定位

- 读写的定位
  - 打开文件读写数据时，需要判断和控制文件的读写位置，如数据是否读完，或者需要读写指定位置上的数据等。
  - 在读写文件时，Matlab 自动创建一个文件位置指针来管理和维护文件读写数据的起始位置。
  - Matlab 提供了几个文件位置指针定位操作函数：
    fseek、ftell 、 frewind 和 feof

fseek

- 读写的定位

```
status=fseek(fid,offset,origin)
```

  - `fid` 为文件句柄
  - `offset` 表示位置指针相对偏移的字节数，若为正表示向文件尾方偏移，若为负表示向文件头方向偏移
  - `origin` 表示位置指针移动的参照位置，有三种取值：
    - `'cof'` 表示当前位置，
    - `'bof'` 表示文件的开始位置，
    - `'eof'` 表示文件末尾；
  - 若定位成功，`status` 返回值为 `0`，否则返回 `-1`

ftell、frewind 和 feof

- 读写的定位

```
position=ftell(fid)
```

- 返回值为从文件开始到指针当前位置的字节数

```
frewind(fid)
```

- 将位置指针返回到文件的起始位置

```
eofstat=ftell(fid)
```

- 判断文件位置指针是否达到文件结束位置，若文件位置指针已在文件末尾，则返回 1，否则返回 0

上机作业

1、编写程序，计算 1 到 225 之间所有整数的平方根，要求将计算结果既在屏幕上输出，也同时将计算结果自动存入文本文件：data91.txt 中。

（程序取名 m91.m）

2、从课程主页上下载二进制数据文件 data09.dat，从文件中读取100个元素（双精度），构成一个 50 X 2 的矩阵。然后将其写入到一个文本文件（data09.txt）中，按两列排放。

（程序取名 m92.m）