

《数学软件》第1讲

Introduction of M-Calculator

主讲教师：胡贤良

浙江大学数学科学学院

暑期短学期课程, 7月6日-7月13日, 2020年

《数学软件》 内容纲要

Octave 是执行数学计算(数值算法)非常好的开源软件。用于计算的脚本(Script)被存储为后缀名.m，我们不妨称其为**M-计算器**。

- 1 M-计算器使用简介
- 2 Thinking in **Matrix** - “万物皆数”
- 3 图像处理
- 4 科学计算
- 5 机器学习
- 6 深度学习
- 7 用 \LaTeX 撰写数学文档
- 8 其他数学计算用软件的简短介绍

本讲内容

- 1 数学软件- 简介
- 2 Octave: M-Calculator
- 3 M-文件(脚本编程)

软件安装

- 基于IDE(如Matlab, Octave自带的桌面环境)的开发是最方便的; 但是学习软件和算法的精髓就是在于折腾!
- 以Win 10 为例, Octave单独一个软件就可以完成本课程所有计算任务。
- 以下的环境是可选的(TeXmacs 或jupyter):
 - ① Python 3.6+
 - ② pip install jupyter
 - ③ configure the environment

Jupyter + Octave

JupyterLab - Mozilla Firefox

tabs: jupyter octave plot, Octave绘图并可可视化, Octave入门-简介, jupyternotebook, Jupyter-Notebook, JupyterLab

address bar: localhost:8888/lab

Menu: File, Edit, View, Run, Kernel, Tabs, Settings, Help

File Explorer: / ... / Math Software (06188220) / code /
Name: lec2_m-calculator.i...

Code Editor: lec2_m-calculator.ipynb

Title: 基于Jupyter的m脚本开发示例

```
[39]: x = (0:20)/20*pi;
```

```
[42]: plot(x,sin(x),'b-*', x, cos(x), 'r-o');grid on;axis tight
```

Plot: A line graph showing two trigonometric functions. The x-axis ranges from 0 to 6, and the y-axis ranges from -1 to 1. A blue line with asterisk markers represents the sine function, and a red line with circle markers represents the cosine function. The plot includes a grid and the axes are labeled.

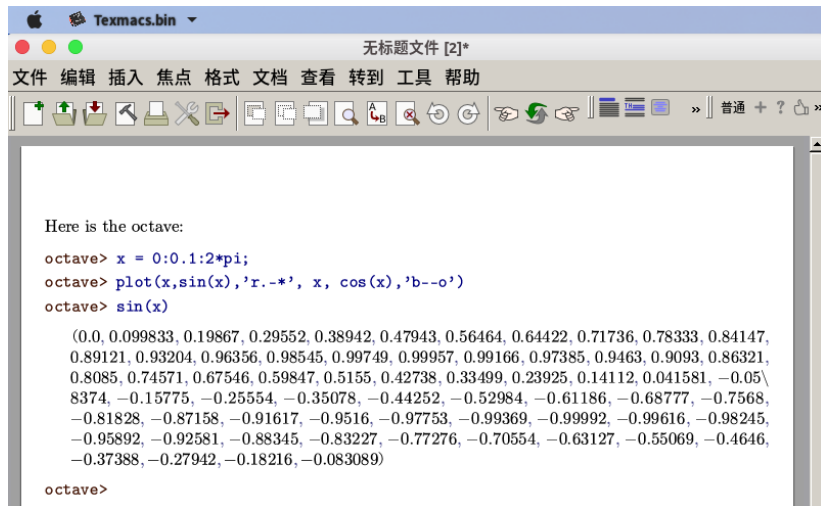
Working with Open Source(开源)

- Jupyter - 可用于学习python,julia,r,octave等“脚本型”计算语言，虽对脚本调试支持比较差，但对于初学入门和对语言十分熟练的用户来说是友好的
- 本次课堂练习和讲解相关的代码将以Jupyter的.ipynb格式和导出的PDF发布，供不同需求读者使用
- Octave 内核是 用于关联Octave应用程序和Jupyter的连接程序，利用pypi安装包是最基本的方式。不同操作系统下安装的难易程度会有所不同：Linux/Mac系统 Windows系统
- 实践过程可能遇到多次失败，需要对kernel重复增删操作。关于Jupyter Kernel的概念，可以参考[这里](#) 和 [这里](#)

反其道而行：排版环境中插入计算环境

"GNU TeXmacs is a free scientific editing platform designed to create beautiful technical documents." — From texmacs官网

- TeXmacs 拥有跟TeX 相同，甚至更好的排版美观程度。
- 拥有超越Word （或者任何一款字处理软件）的，真正的“所见即所得” (WYSIWYG)
- 直接可在屏幕文档里绘图。完全可视化的表格，公式编辑环境。
- 可以嵌入Octave、maxima等计算环境(目前windows下配置困难)

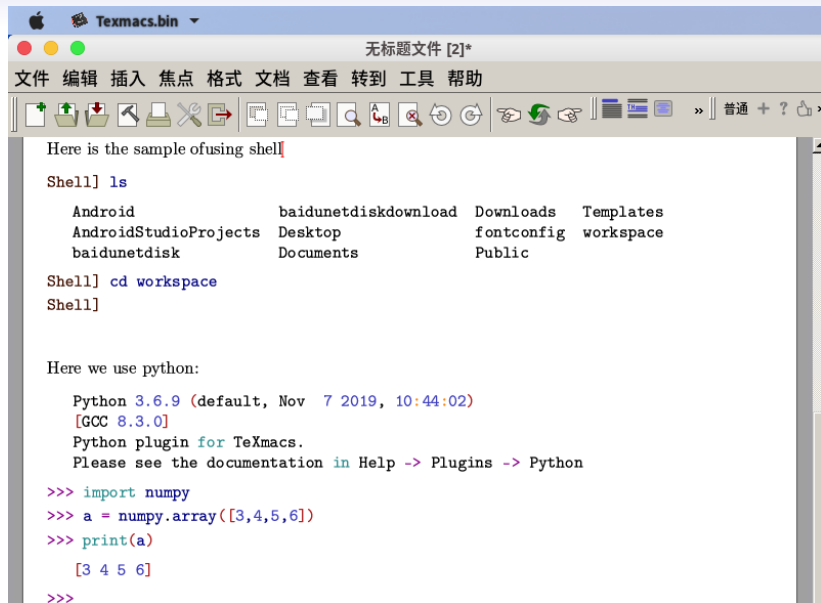


```
Here is the octave:

octave> x = 0:0.1:2*pi;
octave> plot(x,sin(x),'r.-*', x, cos(x),'b--o')
octave> sin(x)

(0.0, 0.099833, 0.19867, 0.29552, 0.38942, 0.47943, 0.56464, 0.64422, 0.71736, 0.78333, 0.84147,
0.89121, 0.93204, 0.96356, 0.98545, 0.99749, 0.99957, 0.99166, 0.97385, 0.9463, 0.9093, 0.86321,
0.8085, 0.74571, 0.67546, 0.59847, 0.5155, 0.42738, 0.33499, 0.23925, 0.14112, 0.041581, -0.05\
8374, -0.15775, -0.25554, -0.35078, -0.44252, -0.52984, -0.61186, -0.68777, -0.7568,
-0.81828, -0.87158, -0.91617, -0.9516, -0.97753, -0.99369, -0.99992, -0.99616, -0.98245,
-0.95892, -0.92581, -0.88345, -0.83227, -0.77276, -0.70554, -0.63127, -0.55069, -0.4646,
-0.37388, -0.27942, -0.18216, -0.083089)

octave>
```



The screenshot shows the TeXmacs application window with the title bar "TeXmacs.bin" and a menu bar with options: 文件, 编辑, 插入, 焦点, 格式, 文档, 查看, 转到, 工具, 帮助. The toolbar contains various icons for file operations, editing, and navigation. The main text area displays the following content:

```

Here is the sample of using shell

Shell] ls

      Android          baidunetdiskdownload  Downloads  Templates
AndroidStudioProjects Desktop          fontconfig workspace
baidunetdisk         Documents          Public

Shell] cd workspace
Shell]

Here we use python:

Python 3.6.9 (default, Nov  7 2019, 10:44:02)
[GCC 8.3.0]
Python plugin for TeXmacs.
Please see the documentation in Help -> Plugins -> Python

>>> import numpy
>>> a = numpy.array([3,4,5,6])
>>> print(a)

[3 4 5 6]

>>>
  
```



Plug-ins for GNU TeXmacs

AboutDownloadLearnContribute

1. GNU Octave

GNU OCTAVE is a high-level language, primarily intended for numerical computations. It provides a convenient command line interface for solving linear and nonlinear problems numerically, and for performing other numerical experiments using a language that is mostly compatible with MATLAB. It may also be used as a batch-oriented language.

OCTAVE has extensive tools for solving common numerical linear algebra problems, finding the roots of nonlinear equations, integrating ordinary functions, manipulating polynomials, and integrating ordinary differential and differential-algebraic equations. It is easily extensible and customizable via user-defined functions written in OCTAVE's own language, or using dynamically loaded modules written in C++, C, Fortran, or other languages.

A first experimental interface between TeXmacs and GNU OCTAVE has been written by MICHAEL GRAFFAM. Unfortunately, we get little help from the main OCTAVE developers for the improvement of the interface and the integration of patches into the mainline of OCTAVE. So, if you like the idea of using OCTAVE from inside TeXmacs, then please help advertising it.

更多参考Octave官网以及一个有趣的人<http://www.yinwang.org/>

1 数学软件- 简介

2 Octave: M-Calculator

3 M-文件(脚本编程)

Octave的功能展示

抽象层面：数值计算、符号计算和数据可视化：

- ① 矩阵分析/计算
- ② 二维、三维数据可视化
- ③ 代数、微分方程求解(精确求解与近似求解)
- ④ 最优化模型求解（数值求解）
- ⑤ 图像分析、信号处理
- ⑥ 数据处理
- ⑦（各种toolbox）

一种专注于科学计算的高级语言？ 一个高级的数学计算器？

例：黄金分割问题

截去一个正方形后仍保持比例的矩形。它满足

$$\frac{1}{\phi} = \frac{\phi - 1}{1} \Leftrightarrow \phi^2 - \phi - 1 = 0.$$

① 数值计算的方法

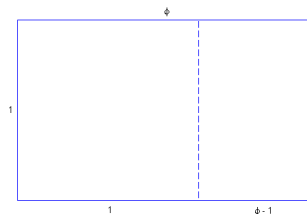
```
p = [1 -1 -1]; r = roots(p);
```

② 符号计算的方法:

```
r = solve('1/x = x-1');
```

③ 视觉计算:

```
f = inline('1/x - (x-1)');  
ezplot(f,[0,4*pi]);
```



特点:语法数学化、编程效率高、易于二次开发。更多参考m-calculator.ipynb

Octave一些常用操作指令

help	帮助系统	exist	变量检验函数
disp	显示变量或内容	echo	信息显示开关
what	目录中文件列表	which	确定文件位置
cd	改变工作目录	dir	现实目录下的文件
path	显示搜索路径	clear	清理内存变量
clc	清除会话窗口	clf	清除图形窗口
save	保存内存变量	load	加载指定变量
who	内存变量列表	whos	内存变量详细信息
hold	图形保持开关	quit	退出MATLAB

变量(Variable)

变量是任何编程语言的基本要素。对于数学用户来说，变量也是一类数学对象：

常用的

- ① 整数
- ② 浮点数
- ③ 复数
- 字符串String
- 元胞数组Cell
- 矩阵(Matrix)与向量(Vector)

等可以认为是数据结构！

变量的命名

- Octave变量无须声明，**用时可随意改变分配空间**；
- 输入格式继承c语言风格；
- 字母打头，由字母、数字和下划线组成，不可有空格和标点
- 区分大小写

其他特殊变量：

inf	eps	bitmax	NaN或nan	i或j	realmin	realmax
pi	beep	ans	nargin	varargin	nargout	varargout

变量的作用域

如在命令行中执行:

```
>> test
```

则Octave系统将寻找其所在的位置, 搜索路径次序为:

- ① 将test视为使用者定义的变量
- ② 若test不是使用者定义的变量, 将其视为常数
- ③ 若test不是常数, 检查其是否是目前**工作目录**下的M文件
- ④ 若不是, 则由搜寻路径寻找是否有test.m的文件
- ⑤ 若仍找不到, 则系统发出提示声并打出**错误提示**信息

基本数据类型之-复数

Octave系统允许使用复数，但须先建立一个复数基，可用

$$i = \sqrt{-1} \quad \text{or} \quad j = \sqrt{-1}.$$

- 建立复数基后，复数可用下面语句输入

$$z = 3 + 4i;$$

- 如下两种输入复数矩阵的方式是等价的

$$A = [1 \ 2; 3 \ 4] + i * [5 \ 6; 7 \ 8]$$

$$A = [1 + 5 * i, 2 + 6 * i; 3 + 7 * i, 4 + 8 * i]$$

- 在MATLAB中复数的四则运算语法和实数一致

数学基本对象-向量的Octave表示

n 维向量被认为是 $n \times 1$ 的矩阵

- 向量生成方法:

- 1 直接输入
- 2 线性等分函数linspace
- 3 对数等分函数logspace

- 向量的基本运算:

- 1 向量加 $\vec{a} + \vec{b}$ 与数加 $a + \vec{b}$
- 2 数乘、内积 $\text{dot}(\vec{a}, \vec{b})$ 或 $\vec{a} \cdot \vec{b}$
- 3 外积 $\text{cross}(\vec{a}, \vec{b})$ 与混合积(注意顺序)

例: 已知向量 $a_1 = [1, 2, 3]^T$, $a_2 = [-1, 4, 0]^T$, $a_3 = [2, 0, 2]^T$,
请分别计算: $2a_1 + 3a_2 - 4a_3$ 、 $a_1 \cdot a_2$ 、 $a_2 \times a_3$ 以及 $a_3 \otimes a_1$.

Example (向量与多项式计算)

- 多项式表示方法

- ① 系数向量直接输入: $p = [1, -5, 6, -33]$; `poly2sym(p)`
- ② 特征多项式: $A = [1, 2, 3; 4, 5, 6; 7, 8, 9]$; `p=poly(A)`
- ③ 由根创建多项式: $v = [-5, -3 + 4i, -3 - 4i]$; `p=poly(v)`

- 多项式运算

- ① 求值: `polyval(p,[1, 2; 2, 1]);`
- ② 求根: `roots(p)`
- ③ 乘除法(即向量卷积): `p = conv(p1,p2); p1 = deconv(p,p2)`
- ④ 微分: `(polyder)`
- ⑤ 拟合: `(polyfit)`

基本数据类型-矩阵

- ① 小矩阵直接输入[1 2 3;4 5 6;7 8 9];
 - ② 大矩阵通过M-文件得到;
 - ③ 某些特殊矩阵预定义（见下一页）
- 参考.ipynb中的示例

典型矩阵- 直接调动

zeros(m,n)	全零矩阵	ones(m,n)	全一矩阵
eye(n)	单位阵	diag(n)	对角阵
rand(m,n)	随机阵	randn	正态分布随机阵
magic	幻方阵	compan	多项式伴随矩阵
gallery	Higham测试阵	rosser	特征值测试矩阵
hilb	Hilbert矩阵	invhilb	反Hilbert矩阵
hankel	Hankel矩阵	toeplitz	Toeplitz矩阵
pascal	Pascal矩阵	vander	范德蒙矩阵
hadamard	Hadamard矩阵	wilkinson	特征值测试矩阵

访问矩阵/向量元素

- 用户可以象对c语言的数组一样访问矩阵或向量中的元素：
 $a(3)$ 表示向量 a 的第3个元素，
 $A(3,2)$ 表示矩阵 A 的第3行、第2列的元素
- 与c数组不同的是：这里矩阵和向量的下标是从1开始计数的
- 除了上述最基本的矩阵元素操作外,还有极为常用的矩阵的抽取、扩展和分解:

$$B = A(2:3, 1:2);$$
$$A(2,:) = [];$$
$$A = [A; \text{zeros}(\text{size}(A,1),1)];$$
$$B = A'; C = \text{transpose}(A);$$

矩阵四则运算

除基本四则运算：加减、乘、左/右除等外：

reshape	变维	flipdim	按维数翻转
flipud	上下翻转	fliplr	左右翻转
rot90	逆时针旋转	diag	取对角向量
tril	提取下三角	triu	提取上三角
det	方阵行列式	inv	方阵的逆
cond	方阵条件数	trace	方阵的迹
norm	求范数或模	rank	矩阵的秩

Example (利用矩阵运算计算面积/体积)

n 维单纯形有 $n + 1$ 个顶点: $\{x_i, y_i, z_i\}_{i=0}^n$ 。则 $n = 2$ 时为三角形:

$$S = \pm \frac{1}{2} \begin{vmatrix} 1 & 1 & 1 \\ x_0 & x_1 & x_2 \\ y_0 & y_1 & y_2 \end{vmatrix}$$

即 $x_1y_2 + x_2y_0 + x_0y_1 - x_2y_1 - x_0y_2 - x_1y_0$. $n = 3$ 时为四面体:

$$V = \pm \frac{1}{6} \begin{vmatrix} 1 & 1 & 1 & 1 \\ x_0 & x_1 & x_2 & x_3 \\ y_0 & y_1 & y_2 & y_3 \\ z_0 & z_1 & z_2 & z_3 \end{vmatrix}$$

- 利用矩阵计算任意给定单纯形的面积(triangle.mat)

线性方程组求解

线性代数方程组是科学计算中最常见的问题之一：求含 n 个分量的列向量 x ，满足

$$Ax = b$$

其中， A 是已知的 n 阶方阵， b 是含 n 个分量的已知列向量。

MATLAB提供的一些求解方法：

- 求逆： $x = \text{inv}(A) * b$
- 左除： $x = A \backslash b$
- $x = \text{linsolve}(A, b)$ —LU或QR分解

关于左除“\”与右除“/”

若 A 为任意大小和形状的矩阵，而矩阵 B 和 A 的行数一样多，那么求解 $AX = B$ 用左除，表示：

$$X = A \backslash B$$

类似地若 A 在右边，如 $XA = B$ ，并且矩阵 B 和 A 的列数一样多，那么可用右除计算：

$$X = B / A$$

练习

- 请求解如下三阶线性方程组

$$\begin{bmatrix} 10 & -1 & 0 \\ -3 & 2 & 6 \\ 5 & -1 & 5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 7 \\ 4 \\ 6 \end{bmatrix}$$

关于排列矩阵(permutation matrix)

$$P = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

- 每行或每列上有且仅有一个1;
- 可以用来表示矩阵行(PA)或列(AP)交换;
- 若向量 $p = [4, 1, 3, 2]$, 则 $P * A$ 和 $A(p, :)$ 在MATLAB中等价;
- 线性方程组 $Px = b$ 的解为 $x = P^T b$

矩阵的其他运算符

算术运算符	$+$ $-$ $*$ $.*$ \wedge $.\wedge$ \backslash $.\backslash$ $/$ $./$
关系运算符	$<$ $<=$ $>$ $>=$ $==$ $\sim=$
逻辑运算符	$\&$ (与) $ $ (或) \sim (非)

:	截取向量的指定部分	%	注释
;	矩阵分行、屏蔽显示	=	赋值
!	从MATLAB调用系统命令	,	分隔矩阵列
'	矩阵转置、共轭、字符串	...	续行
()	函数调用和制定运算顺序	[]	矩阵操作

函数(Function)

- ① 系统功能函数(如打印变量、绘图、外接API)
- ② 数学库函数
- ③ 自定义内联(inline)函数
- ④ 自定义.m 文件函数

内建数学函数- 直接调动

$\sin(x)$	x 的正弦	$\operatorname{asin}(x)$	x 的反正弦
$\cos(x)$	x 的余弦	$\operatorname{acos}(x)$	x 的反余弦
$\tan(x)$	x 的正切	$\operatorname{atan}(x)$	x 的反正切
$\log(x)$	x 自然对数	$\log_2(x)$	以2为底对数
$\operatorname{round}(x)$	x 四舍五入	$\log_{10}(x)$	x 以10为底对数
$\operatorname{floor}(x)$	向左取整	$\operatorname{ceil}(x)$	向右取整
$\operatorname{sqrt}(x)$	\sqrt{x}	$\exp(x)$	e^x
$\operatorname{abs}(x)$	$ x $	$\operatorname{angle}(x)$	复数的相角
$\operatorname{real}(x)$	取 x 的实部	$\operatorname{imag}(x)$	取 x 的虚部
$\operatorname{conj}(x)$	复数的共轭	$\operatorname{mod}(x, y)$	x/y 的余数
$\operatorname{gcd}(x, y)$	最大公约数	$\operatorname{lcm}(x, y)$	最小公倍数

特殊函数

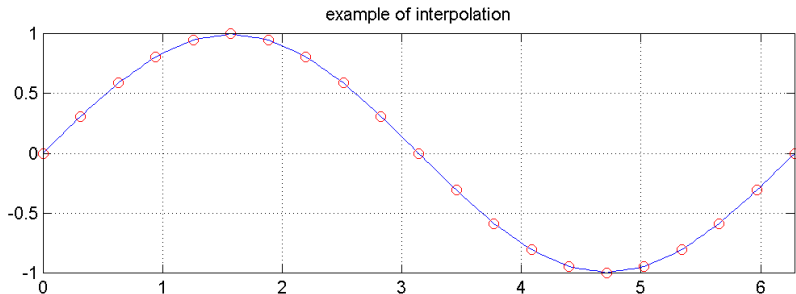
besselj	第一类Bessel函数	erf	误差函数
bessely	第二类Bessel函数	legendre	Legendre函数
besselh	第三类Bessel函数	beta	Beta函数
besseli	改进第一类Bessel函数	gamma	Gamma函数
besselk	改进第二类Bessel函数	ellipj	jacobi椭圆函数

插值问题

已知平面上 n 个数据点 $\{(x_k, y_k)\}_{k=0}^n$ (polyfit_data.mat), 找一个函数 $P(x)$ 满足

$$P(x_k) = y_k, \quad \forall k = 0, \dots, n.$$

几何意义: 找一条曲线 $y = P(x)$ 通过给定的型值点。

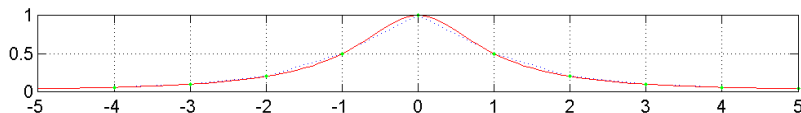


分段插值interp1

```
1 yi = interp1(x,y,xx,'nearest');
2 plot(xx,yi,'b:',xx,yh,'r',x,y,'g.');
```

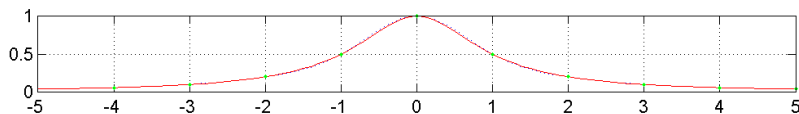
3 axis equal; axis([-5 5 0 1]); grid on;

4 print -dpng runge1.png; pause;



样条插值(spline)

```
1  ys = spline(x,y,xx);  
2  plot(xx,ys,'b:',xx,yh,'r',x,y,'g. ');  
3  axis equal; axis([-5 5 0 1]); grid on;  
4  print -dpng runge2.png;
```



方程求根问题

$$\text{求方程组} \begin{cases} \sin(x_1) + x_2 + x_3^2 e^{x_1} - 4 = 0 \\ x_1 + x_2 x_3 = 0 \\ x_1 x_2 x_3 = -2 \end{cases} \text{ 的近似解:}$$

$$[x, fval] = fsolve('group1', [1, 1, 1]);$$

即能得到一个数值解，其中group1函数需要自己提供：

```
1 function f = group1(x)
2     f = [sin(x(1)) + x(2) + x(3)^2*exp(x(1)) - 4; ...
3         x(1) + x(2)*x(3); ...
4         x(1)*x(2)*x(3) + 2];
```

此外，还有提供一元非线性方程零点求解的fzero。

1 数学软件- 简介

2 Octave: M-Calculator

3 M-文件(脚本编程)

M-文件

我们可以在命令窗口中完成大部分问题的计算，然而科学计算的精神在于能重现数值结果。这就需要将一系列的指令保存成序列以便重复执行，这就是**程序**。在MATLAB中：

- 程序也称为脚本，以M-文件（后缀名为.m）形式保存；
- M-文件通常保存在当前目录、默认目录或系统目录下；
- 可以直接在命令窗口中输入M-文件的名称来运行该脚本；
- 软件系统执行命令的检查顺序：

变量→系统函数→自定义函数(M-文件)

基本程序结构

类似于c/c++/java等其他高级计算机语言，

- 提供了三种基本的程序结构：
 顺序结构、循环结构、选择结构；
- 其他流程控制语句，如：continue、break、return等；
- 程序一般包含三个部分：
 数据输入、功能处理和**结果输出**。

循环结构

循环是程序设计最根本的目的，是程序工作最基本的方式！

- 有限次循环结构:

```
for n = n1:step:n2
    command recycled;
end
```

- 条件式循环结构:

```
while (conditions)
    command recycled;
end
```

- ① 根据 $e \sim 1 + 1 + \frac{1}{2!} + \frac{1}{3!} + \cdots + \frac{1}{n!}$ 求 e 近似值，精确到 10^{-8}
- ② 求正整数 $n (n \leq 20)$ 的阶乘: $p = n! = 1 \times 2 \times 3 \times \cdots \times n$

1.7.1 for 循环

例如:

循环可以嵌套

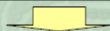
```
for i=1:5;
x(i)= 2 * i;
end
```



x =

2 4 6 8 10

```
for i=1:5;
    for j=1:3
        x(i , j)= i * j;
    end
end
```



x =

1	2	3
2	4	6
3	6	9
4	8	12
5	10	15

Figure: For循环例子

下面用一个简单的问题来说明 **while** 循环的用法。求解阶乘 $n!$ 具有100位数字的整数 n 是多少？

```
n=1;
while prod ( 1 : n ) < 100;
n=n+1;
end
n
```



结果为 $n=5$ 。

$5 \times 4 \times 3 \times 2 \times 1 = 120$

若 $n=6$ ，则 $6! = 720$ ，非100位数字

while 循环的一般格式为：

```
while 表达式
    语句组
end
```



Figure: While循环例子

选择结构I

- 单项选择判断结构:

```
if (condition)
    do-command1;
else
    do-command2;
end
```

- 多项选择判断结构:

```
if (condition1)
    do-command1;
elseif (condition2)
    do-command2;
else
```

1.7.2 if 语句

```
if 条件语句
    表达式1
else
    表达式2
end
```

```
if 条件语句1
    表达式1
else if 条件语句2
    表达式2
else
    表达式3
end
```

例如:

$$F = \begin{cases} 10 & t \leq 0 \\ 20 & 0 < t \leq 1 \\ 30 & 1 < t < 2 \\ 40 & t \geq 2 \end{cases}$$



```
if t >= 2
    F = 40
else if t > 1
    F = 30
else if t > 0
    F = 20
else
    F = 10
end
```



Figure: If条件判断例子

选择结构II

```
method = 'Bilinear';  
switch lower(method)  
    case 'linear','bilinear'  
        disp('Method is linear')  
    case 'cubic'  
        disp('Method is cubic')  
    case 'nearest'  
        disp('Method is nearest')  
    otherwise  
        disp('Unknown method.')  
end
```

最后的输出是: **Method is linear**

其它程序控制语句

- break: 结束当前层的循环;
- continue: 当前循环中剩余命令不执行, 进入下一个循环;
- return: 函数或命令结束, 返回上一层调用
- pause: 程序在此处暂停直到有键盘响应;
也可以用pause(n)指定暂停的秒数。

二分法

❶ 在 $[1, 2]$ 中找到 $\sqrt{2}$ 的近似值;

```
1 k = 0;
2 while abs(b-a) > eps*abs(b)
3     x = (a+b)/2;
4     if (sign(f(x)) == sign(f(b)))
5         b = x;
6     else
7         a = x;
8     end
9     k = k + 1;
10 end
```

● 二分搜索尽管收敛很慢，但是一个稳定、有效的算法

Example (Newton迭代法)

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$

求 $x^3 - 3x - 1 = 0$ 在 $x = 2$ 附近的根, $x_0 = 2, x_1 = 1.9$

param		100	200	400	800
MEM(Mb)		13	26	51	103
jk	cpu	0.1	0.07	0.033	0.015
	rft	0.12	0.09	0.021	0.01

Example (割线法(Secant Method))

Newton迭代法每一步要计算 $f'(x)$ ，这在函数不太光滑或多元情形十分不方便，很多时候如下所谓的割线法：

$$x_{k+1} = x_k - \frac{f(x_k)(x_k - x_{k-1})}{f(x_k) - f(x_{k-1})}$$

... do it with m script

Example

求解下列方程组

$$x_1 - 0.7 \sin x_1 - 0.2 \cos x_2 = 0 \quad (1)$$

$$x_2 - 0.7 \cos x_1 + 0.2 \sin x_2 = 0 \quad (2)$$

在(0.5, 0.5)附近的解，精度要求 10^{-3}

第一个结果是关于使用符号计算程序solve和数值计算程序fsolve的区别，采用例子(2) 方程组

$$\begin{cases} \sin(x_1) + x_2 + x_3^2 e^{x_1} - 4 = 0 \\ x_1 + x_2 x_3 = 0 \\ x_1 x_2 x_3 = -2 \end{cases}$$

解析解solve过于复杂而非必要。可以用fsolve获得近似解：

$$[x, fval] = fsolve('group1', [1, 1, 1]);$$

solve() V.S. fsolve()

函数名/方法	时间消耗(s)	得到的解
solve	0.52	$x = \sqrt{2}$ $y = -1.37$ $z = 0.13$
fsolve	0.012	$x = 1.414$ $y = -1.372$ $z = 0.1322$

Table: 解析解与数值解的异同

当初始选取不同的时候，数值解得到的结论稍有不同...

Newton迭代的m-script

```

1  function x = newton(x0,TOL,MAXIter)
2  k = 1;
3  residual = 100;
4  while ( residual > TOL && k <= MAXIter)
5  x = x0 - fun_df(x0)\fun_f(x0);
6  residual = norm(x - x0);
7  fprintf (' The residual at step %d is ...
8  %20.16f\n', k, residual );
9  x0 = x;          k = k+1;
10 end
11
12 function f = fun_f(x)
13 % f = 3*x^2 - exp(x);
14 f = [ x(1) - 0.7*sin(x(1)) - 0.2*cos(x(2));
15 x(2) - 0.7*cos(x(1)) + 0.2*sin(x(2))];

```

数值计算

针对算例??运用newton迭代:

```
>> x = newton([0; 100], 1e-10, 120);
```

得到的输出如下

The residual at step 1 is 89.1156691627774364

The residual at step 2 is 23.2238260567751880

The residual at step 3 is 12.1656181120608977

The residual at step 4 is 5.9753216737702886

The residual at step 5 is 0.8255696336886729

The residual at step 6 is 0.1658865655257854

The residual at step 7 is 0.0133929915669240

The residual at step 8 is 0.0000781732021693

The residual at step 9 is 0.0000000025337888

Final Comment

“Well-written programs are better than badly-written ones. They have fewer errors and are easier to **debug** and **modify**. It is important to think about style from the beginning.”

—Brian Kernighan