数值计算方法

杨扬

第一讲： Introduction（绪论）

Chapter 0.3 函数

matlab的内建函数，比如“plot”，“sqrt”，“sin”...

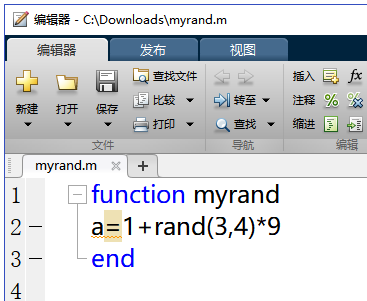
* 内建函数，也可以称为指令

例子：rand 产生0到1间平均分布的随机数

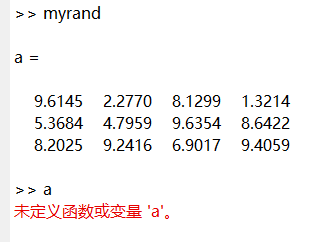
>>rand(3,4)

>>2+rand(3,4)\*5

如何定制需要的函数？



* 函数的输入变量与输出变量



函数有自己私有的临时workspace，函数中的变量存储在私人workspace，运算结束该空间被删除。

存储于这个空间的变量叫局域变量（local variable）。

* 如何把这些变量输出到外部的workspace？

function myrand → function a=myrand

* 假如要修正函数中的公式怎么办？例如 a=x+rand(3,4)\*y, x,y是我们根据需要设定的参数。

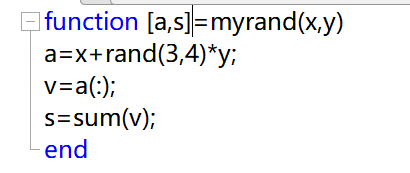
function a=myrand → function a=myrand(x,y)

目前 两个输入变量，标量

一个输出变量， 矩阵

* 假如要输出多个变量怎么办？

比如输出a矩阵外，同时输出a所有元素的和。



>>myrand(x,y) 语句没写全发生什么？只输出第一个变量结果。

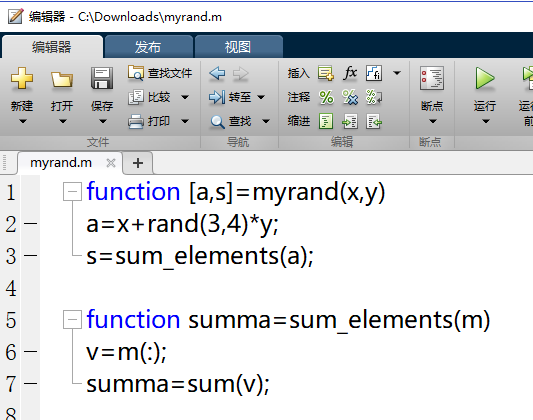
* 函数的正式定义

function [out\_arg1, out\_arg2,...]=function\_name(in\_arg1, in\_arg2, ...)

1. 函数名最好跟函数的作用有关
2. 不要使用内建函数名，如， plot， sqrt

* 子函数

如果我们考虑的问题很复杂，我们很可能将函数分为很多个，这些函数可以放在同一个m文件中。

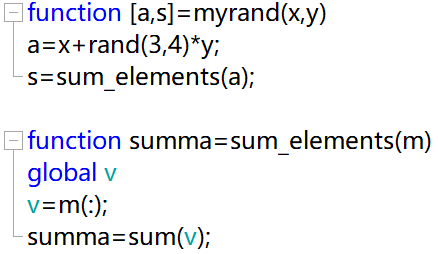


第一个函数被称为主函数。第二个被称为子函数或者局域函数，只能在主函数中被调用，不能从外部调用。

* Scope 局域变量 全局变量

局域变量只能从函数内部访问

>>v?



“global”定义全局变量

>> global v; >>[a,s]=myrand(1,2); >>v?

Warning: 初学者不建议随意使用全局变量，因为任何函数都能修改全局变量的值，容易出错。

* 函数的优势

1. 复杂问题简化成若干小问题，易于操作
2. 代码重复利用
3. 具有解决一类问题的能力。比如，plot只要调整参数就能画各 种图像。
4. 一组函数可以构成函数库，解决更多问题。开源函数库，商业 化的函数库。

Chapter 0.4 Matlab控制流（control flow）

要求：掌握c语言中已经熟悉“if、 for、 while”等指令控制结构；逻辑运算符号；理解预分配。

* if 控制

例子

function guess(x)

if x==2

fprintf(“that’s right \n”)

end

if开头，紧跟逻辑判断，end结束

* If-else控制

function guess(x)

if x==2

fprintf(“that’s right \n”)

else

fprintf(‘error! \n’)

end

* if-elseif控制function guess(x)

if x==4

fprintf(‘that’s right \n’)

elseif x<4

fprintf(‘too small! \n’)

else

fprintf(‘too large! \n’)

end

* 再多一些

if x==2

fprintf(‘that’s right \n’)

elseif x==3

fprintf(‘... \n’)

elseif x==4

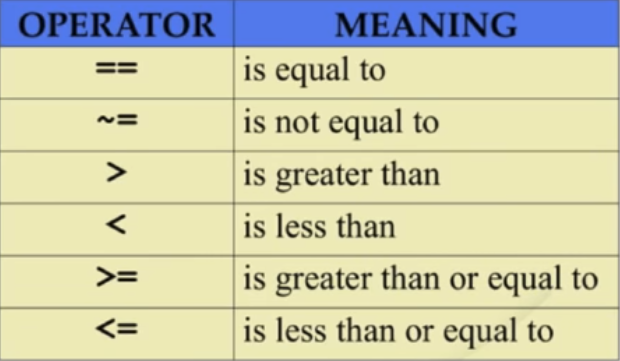
fprintf(‘... \n’)

elseif x==5

....

end

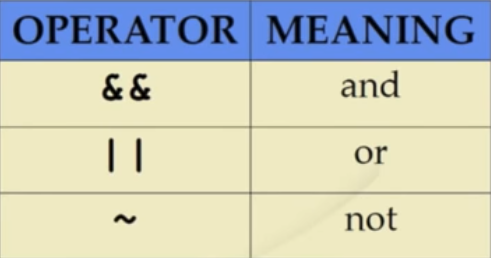
* 逻辑判断运算符



>>3\*5>=14 ?

>> [5,1,3,4,5]<[1,3,43,2,35] ?

>> [5,1,3,4,5]<4 ?



function a=order3(x,y,z)

if x<=y && y<=z

a=1;

elseif x>=y &&y>=z

a=-1;

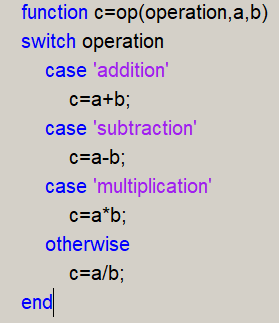
else

a=0;

end

* Switch,case,otherwise

Execute one of several groups of statements

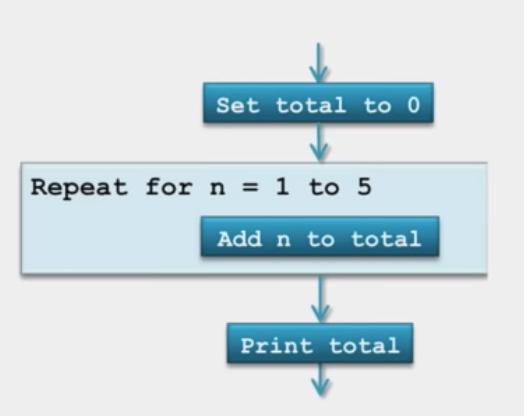


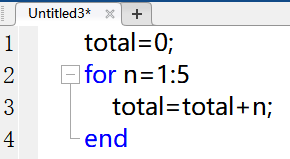
* for循环 （for loops）

例子： 一个简单的循环

>>n=1:5;

>>total=sum(n);





如何写一个function文件，求任意整数的阶乘？

* While循环

while *control statement*

*expression block*

end

例子：

function [n total]=possum(limit)

total=0;

n=0;

while total <=limit

n=n+1;

total = total+n;

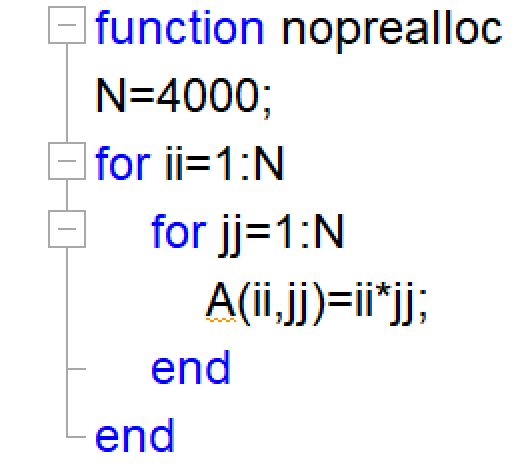
end

fprintf('sum: %d count: %d \n',total,n)

* 预分配（preallocation）

>> tic;sum(1:1e3);toc

Elapsed time is 0.004766 seconds.



>>tic; nopreallic;toc

如何改进？ for循环前预分配 ，插入语句“A=zeros(N,N);”

没有预分配，

每执行一行for循环

必须重新在内存中重新找一个位置

将整个矩阵的内容复制过去

再执行循环中的语句.

预分配 先在内存中分出匹配矩阵的位置 节省分配内存的时间。

matlab会提示你没有进行合适的内存分配

要求：掌握利用matlab构造矩阵；输入/输出的控制；调试器。

* 多态性

输入变量可以是标量、矩阵、矢量等不同类型，比如，sqrt。

内建函数多具备多态性。

* 构造矩阵

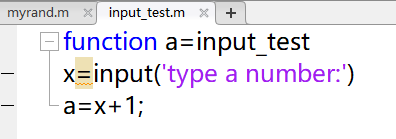
>>a=[1:3;4:6]; >>zeros(2,3); >>ones(3,1); >>eye(3);

>>b=diag([1,2,3,4]); >>rand(3)

每次matlab启动后，运行rand，结果总是0.8147 （所 有的随机数产生器都是赝随机数）

* input/output

1. 输入



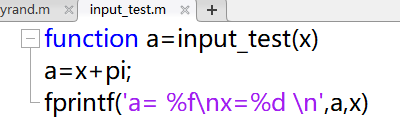
>>input\_test

2.输出

>>fprintf(‘jungle jungle ocean \n’)

这里“\n”表示另起一行 new line

>>fprintf(‘jungle jungle ocean \n\n\n’)



按照顺序，字符串中%f表示变量a，%d表示变量x。

%d 科学计数法表示变量的值

%f 正常十进制数，可以表示小数，%.2f 显示两位小数

作业：编写计算exp（x）的函数, 计算结果与matlab内置的exp函数计算结果比较差别

Chapter 0.6 matlab中的基本数值计算

要求：能够利用matlab求解几种常见的计算问题。求导；积分；非线性方程的零根；函数的极小值；常微分方程的解；矩阵的本征值和本征态

* 求导

在Matlab的数值计算中，没有专门的求极限和求导指令；由于数值精度有限，应谨慎使用。

求极限

x=1e-5

L1=(1-cos(2\*x))/(x\*sin(x))

L2=sin(x)/x

理论结果：



* 【例】已知 ， 求在区间 中的近似导函数。

d=pi/100;

t=0:d:2\*pi;

x=sin(t);

dt=d;

x\_eps=sin(t+dt);

dxdt\_eps=(x\_eps-x)/dt;

plot(t,x)

hold on

plot(t,dxdt\_eps)

* 差分： >>dx=diff(X)；
* 梯度：>>FX=gradient(F)；>>[FX,FY]=gradient(F)

t=0:d:2\*pi;x=sin(t);

dxdt\_diff=diff(x)/d; dxdt\_grad=gradient(x)/d

plot(t,x,'b'),hold on

plot(t,dxdt\_grad,'r')

* 积分

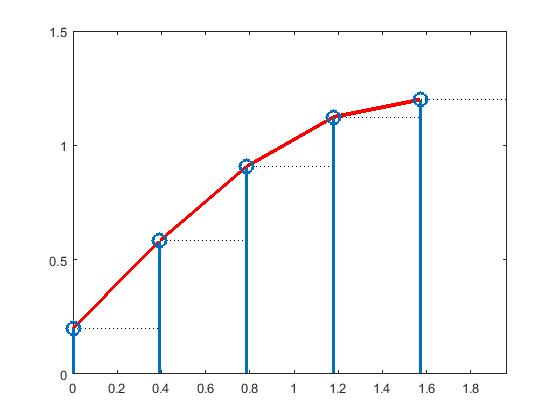
例子：计算

最笨的办法

d=pi/8; t=0:d:pi/2;

y=0.2+sin(t);

s=sum(y); s\_sa=d\*s; s\_ta=d\*trapz(y);



matlab内建的积分函数使用优化的Simpson算法，可以控制精度。

>>S1=quad(fun,a,b,tol) %一重积分

例子

fx='exp(-x.^2)';

I=quad(fx,0,1,1e-8)

p.s. 为什么函数是数组计算的格式？Matlab积分分成三步，1.根据优化的算法将积分上下限分割成若干个点，构成一个数组；2.这一数组代入被积函数；3.最后根据积分算法求得积分的值。第二步和第三步是数组计算。

>>S2=dblquad(fun,xmin,xmax,ymin,ymax,tol) %二重积分

例子

s=dblquad('x.^y',0,1,1,2)

>>S3=triplequad(fun,xmin,xmax,ymin,ymax,zmin,zmax,tol); %三重积分

* 函数句柄 （function handle）

A function handle is a data type that stores an association to a function.

包含了函数的路径、函数名、类型以及可能存在的重载方法.

* 构造函数句柄

>>f=@function\_name

例子：function y = computeSquare(x)

y = x.^2;

end

>>f = @computeSquare;

>>a = 4;

>>b = f(a);

利用函数句柄，使用quad做积分

quad(@fun,a,b,tol)

>>quad(@computeSquare,0,2);

>>quad(f,0,2);

* 匿名函数（anonymous functions）
* Inline

匿名函数只包含一行表达式，不需保存为函数文件形式。

@(arglist)function\_expression

例子： @(n)n.^2

@(x,y)x.^2+y

>>quad(@(n)n.^2,0,2)

>>dblquad(@(x,y)x.^2+y,0,1,1,2)

* 寻找非线性方程的零根

Matlab命令：fzero(@fun,x0) %寻找函数fun在x0附近的零根

例子：计算的数值解。

问题等价于计算的正数解。估算，这个问题的解介于1 和2之间

>>fzero(‘x^2-2’,1)；

>>fzero(‘x^2-2’,2);

p.s. 试一试将函数写为‘x.^2-2’?

例子：f(X)=X^3-2X-5的根

function y = f(x)

y = x.^3-2\*x-5;

>>fzero(@f,2)

例子：fun = @cos; % function

x0 = [1 2]; % initial interval

x = fzero(fun,x0)

Matlab命令：fsolve %求解非线性方程、方程组

function F=myfun(x)

F=[2\*x(1)-x(2)-exp(-x(1));

-x(1)+x(2)-exp(-x(2))];

>>x0=[-5;-5];

>>[x,fval]=fsolve(@myfun,x0)

* 寻找函数的极小值

寻找函数的极小值是计算非线性方程零根的应用。为什么？

[x,fval,exitflag,output]=fminbnd(@fun,x1,x2) %在x1和x2之间寻找函数fun的极小值。

[x,fval,exitflag,output]=fminsearch(@fun,x0) %在x0附近寻找函数fun的极小值。

x:matlab 返回的极小值

fval：极小值代入函数的结果

exitflag：命令停止运行的原因，益整数形式返回

1. 1 函数收敛于解 x
2. 0 迭代次数超出范围
3. -1 由输出函数停止
4. -2 边界不一致，x1>x2 %fminsearch 不包含这一项

Output: 有关优化过程的信息，包括迭代次数，函数计算次数，算法，退出消息。

例子：

[>>f=@(x)x.^2](mailto:%3e%3e@(x)x.%5e2);

>>[x,fval,exitflag,output]=fminbnd(f,-2,2);

>>[x,fval,exitflag,output]=fminsearch(f,0.5);

无论哪种命令算法，matlab只能找到局域的最小值，因此大多数情

况下我们需要做些估计。

>>edit find\_min

x1=-10;x2=10;

yx=@(x)(sin(x)^2\*exp(-0.1\*x)-0.5\*sin(x)\*(x+0.1));

[xn0,fval,exitflag,output]=fminbnd(yx,x1,x2);

xn0,fval

%观察图像

xx=-10:pi/200:10;

yxx=(sin(xx).^2.\*exp(-0.1.\*xx)-0.5\*sin(xx).\*(xx+0.1))

plot(xx,yxx)

%重设搜索区间：

x1=6;x2=10;

[xn00,fval,exitflag,output]=fminbnd(yx,x1,x2);

xn00,fval

例子2：

>>edit find\_min2

ff=@(x)(100\*(x(2)-x(1).^2).^2+(1-x(1)).^2);

x0=[-5,-2,2,5;-5,-2,2,5]; %提供4个搜索起点

[sx,sfval,sexit,soutput]=fminsearch(ff,x0);

sx, disp([ff(sx(:,1)),ff(sx(:,2)),ff(sx(:,3)),ff(sx(:,4))])

* 常微分方程的解

Matlab中解常微分方程的命令有多种，这些命令的调用大同小异，有的只是背后算法的微妙区别。这里我们仅对一种常用命令举例。

[t,y] = ode45(odefun,tspan,y0)

%odefun 常微分方程； tspan 时间区间 y0 初始条件

常微分方程分为初值问题、边界问题和本征值问题。

初值问题的解法最简单，边界问题要复杂，本征值问题更复杂。后两类问题可以利用初值问题的算法改进进行求解。

我们现在只考虑初值问题。

例子：y’=2t 时间区间 [0,5], 初始条件 y0=0.

>>tspan=[0 5];

>>y0=0;

>>[t,y]=ode45(@(t,y)2\*t,tspan,y0)

>>plot(t,y,’-o’)

这个例子太简单，本质上还是积分。

例子：简谐振子

弹簧和小球构成的简谐振子，弹性系数为k，小球质量为m，动力学方程为 简单起见，我们设。

Matlab只能求解线性的常微分方程，对于高阶微分方程，我们需要将其化为线性微分方程组。

令,

于是 。

>>edit dydt

function ydot=dydt(t,y)

ydot=[y(2);-y(1)]

>>edit example

tspan=[0,30]

y0=[1;0]

[tt,yy]=ode45(@dydt,tspan,y0);

plot(tt,yy(:,1));

hold on

plot(tt,yy(:,2));

如果我们将上面这个例子变得复杂些，加入阻尼，

为阻尼系数，我们设为0.2

function ydot=dydt(t,y)

ydot=[y(2);-y(1)-0.2\*y(2)];

如果我们将上面这个例子变得更复杂，不仅考虑阻尼，还额外引入一个驱动力，为驱动频率，我们设为1.2，振幅A我们设为0.1

function ydot=dydt(t,y)

ydot=[y(2);-y(1)-0.2\*y(2)+0.1\*sin(1.2\*t)];

* 矩阵的本征值和本征态（本征矢量）

Matlab 命令：eig

[v,d]=eig(A), v是由本征态构成的变换矩阵，d是矩阵A的本征值构成的对角矩阵。

v矩阵的第n列对应d矩阵第n个对叫元的本征态。

v\*d\*v‘=A