華中科技大學

# 电气工程建模与仿真

## MATLAB编程

电气工程建模与仿真课程组

2022年03月

# 目 录

- 数值微分
- 数值积分
- 微分方程解析解求解
- 常微分方程数值求解
- 偏微分方程数值求解
- 课堂练习

# 目 录

- **数值微分**
- 数值积分
- 微分方程解析解求解
- 常微分方程数值求解
- 偏微分方程数值求解
- 课堂练习

# 数值微分

➢ 微分和导数密切相关，MATLAB与微分相关的函数主要有三个，如下表所示。

| | |
|---|---|
| diff | 差分和近似导数 |
| gradient | 数值梯度 |
| del2 | 离散拉普拉斯算子 |

当已知函数表达式时，diff(y, n)可以求解函数y的n阶导数。**要使用diff求解函数导数，需要定义符号变量**，右边给了一个简单示例：

```
syms x          %定义符号变量
y = x^2+cos(x);  %函数表达式
diff1_y = diff(y,1);  %一阶导数
diff2_y = diff(y,2);  %二阶导数
```

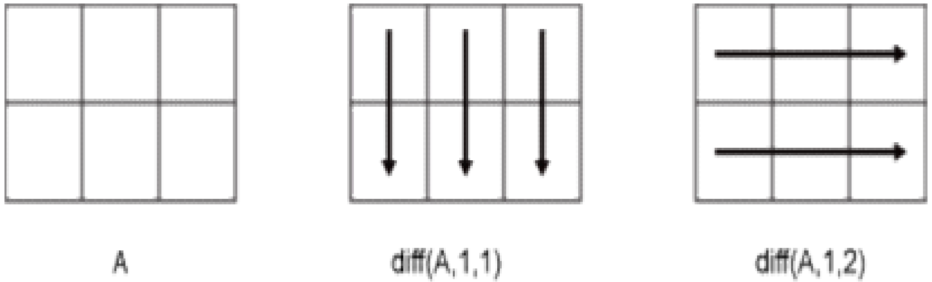
# diff函数

➢ 引用格式如下：

Y = diff(X)

Y = diff(X,n)

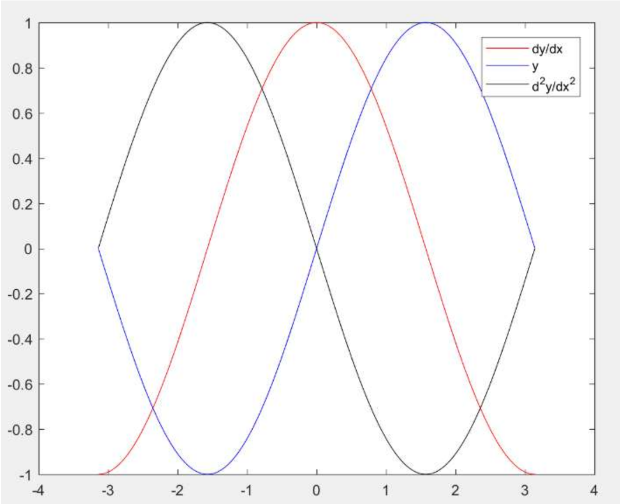
Y = diff(X,n,dim)




□ 示例：

计算差分函数f=sin(X)导数的近似值：

```
h = 0.001;       % step size
X = -pi:h:pi;    % domain
f = sin(X);      % range
Y = diff(f)/h;   % first derivative
Z = diff(Y)/h;   % second derivative
plot(X(:,1:length(Y)),Y,'r',X,f,'b', X(:,1:length(Z)),Z,'k')
```




https://ww2.mathworks.cn/help/matlab/ref/diff.html#btwmxq8-10

# gradient函数

## ➢ 算法:

计算内部数据点的中心差分。例如，考虑一个包含单位间距数据的矩阵 A，它具有水平梯度 G = gradient(A)。内部梯度值 G(:,j) 为：G(:,j) = 0.5*(A(:,j+1) - A(:,j-1)); j = 2 : N-1 之间变化，其中 N = size(A,2)。

gradient 使用单侧差分计算沿矩阵边的值：G(:,1) = A(:,2) - A(:,1); G(:,N) = A(:,N) - A(:,N-1);

## ➢ 语法:

FX = gradient(F)

[FX,FY] = gradient(F)

[FX,FY,FZ,...,FN] = gradient(F)

[___] = gradient(F,h)

[___] = gradient(F,hx,hy,...,hN)

https://ww2.mathworks.cn/help/matlab/ref/gradient.html

## ➢ 示例:

计算 $xe^{-x^2-y^2}$ 在网格上的二维梯度，并绘制向量场的等高线图：

```
x = -2:0.2:2;
y = x';
z = x .* exp(-x.^2 - y.^2);
[px, py] = gradient(z);
figure
contour(x,y,z)    %绘制等高线
hold on
quiver(x,y,px,py)  %绘制向量图
```

# del2函数

## 算法:

$$L_{ij} = \left[ \frac{\left(u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1}\right)}{4} - u_{i,j} \right].$$

$$L = \frac{\Delta U}{2N} = \frac{1}{2N}\left(\frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2} + \frac{\partial^2 U}{\partial z^2} + \ldots\right),$$

$$L = \frac{\Delta U}{4} = \frac{1}{4}\left(\frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2}\right).$$

## 语法:

```
L = del2(U)
L = del2(U,h)
L = del2(U,hx,hy,...,hN)
```

[https://ww2.mathworks.cn/help/matlab/ref/del2.html](https://ww2.mathworks.cn/help/matlab/ref/del2.html)

## 示例:

计算并绘制多变量函数的离散拉普拉斯算子

```
[x,y] = meshgrid(-5:0.25:5,-5:0.25:5);
U = 1/3.*(x.^4+y.^4); h = 0.25; L = 4*del2(U,h);
figure
surf(x,y,L)
grid on
h=title('Plot of $\Delta U(x,y) = 4x^2+4y^2$');
set(h,'Interpreter','latex')
xlabel('x'), ylabel('y'), zlabel('z'), view(35,14) ;
```

# 目 录

- 数值微分
- **数值积分**
- 微分方程解析解求解
- 常微分方程数值求解
- 偏微分方程数值求解
- 课堂练习

# 数值积分

- 积分分为不定积分（int函数）和定积分，**很多积分问题是没有解析解的或者解析解的求解过程非常复杂**，需要采用数值积分MATLAB与数值积分相关的函数主要分为两类，包括**求函数表达式的积分和求数值数据的积分**

| | |
|---|---|
| integral | 数值积分 |
| integral2 | 对二重积分进行数值计算 |
| integral3 | 对三重积分进行数值计算 |
| quadgk | 以自适应高斯-勒让德积分法计算数值积分 |
| quad2d | 计算二重数值积分 - tiled 方法 |
| trapz | 梯形数值积分 |

# integral和quadgk函数

## ➢ 引用格式如下:

q = integral(fun,xmin,xmax)

q = integral(fun,xmin,xmax,Name,Value)

## □ 说明:

q = integral(fun,xmin,xmax) 使用**全局自适应积分**和**默认误差容限**在 xmin 至 xmax 间以数值形式为函数 fun 求积分。

q = integral(fun,xmin,xmax,Name,Value) 指定具有一个或多个 Name,Value 对组参数的其他选项，可以指定绝对误差容限、相对误差容限、数组值函数标志、积分路点等。

## □ 示例:

创建带有一个参数 c 的函数 $f(x)=1/(x^3-2x-c)$。在 c=5 时，计算从 x=0 至 x=2 的积分。

```
fun = @(x,c) 1./(x.^3-2*x-c);
q = integral(@(x)fun(x,5),0,2)
```

**integral的具体用法:**

https://ww2.mathworks.cn/help/matlab/ref/integral.html#btdd9y9

**quadgk的具体用法:**

https://ww2.mathworks.cn/help/matlab/ref/quadgk.html

# integral2、quad2d及integral3函数

## ➢ 引用格式如下：

q = integral2(fun,xmin,xmax,ymin,ymax)

q = integral2(fun,xmin,xmax,ymin,ymax,Name,Value)

## □ 示例：

使用参数 a=3 和 b=5 创建匿名的参数化函数 $f(x,y)=ax^2+by^2$。

```
a = 3; b = 5;
fun = @(x,y) a*x.^2 + b*y.^2;
format long
q = integral2(fun,0,5,-
5,0,'Method','iterated',...    %...表示换行
'AbsTol',0,'RelTol',1e-10)
```

**integral2的具体用法：**
https://ww2.mathworks.cn/help/matlab/ref/integral2.html

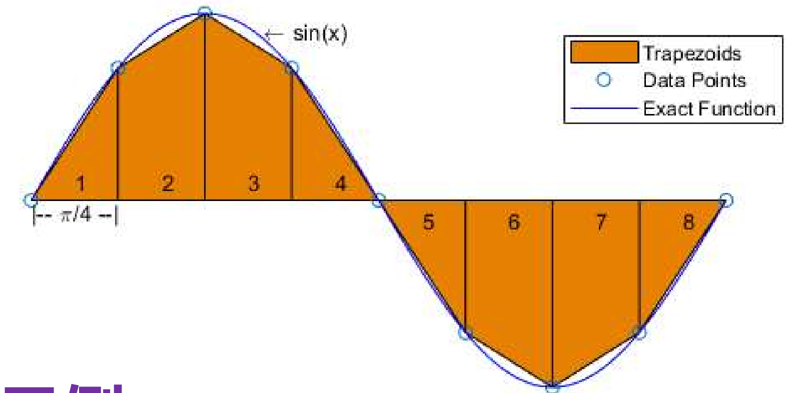
**quad2d的具体用法：**
https://ww2.mathworks.cn/help/matlab/ref/quad2d.html

**integral3的具体用法：**
https://ww2.mathworks.cn/help/matlab/ref/integral3.html

# trapz函数

➤ 当函数未知时，对于离散数据可采用**trapz**函数（梯形法）进行数值积分

$$\int_a^b f(x)dx \approx \frac{1}{2}\sum_{n=1}^{N}(x_{n+1}-x_n)\left[f(x_n)+f(x_{n+1})\right],$$




## ➤ 语法：

Q = trapz(Y)

Q = trapz(X,Y)

Q = trapz(___,dim)

## ➤ 示例：

对具有非均匀数据间距的矩阵的行求积分。

```
X = [1 2.5 7 10];
Y = [5.2   7.7   9.6   13.2;
       4.8   7.0  10.5   14.5;
       4.9   6.5  10.2   13.8];
Q1 = trapz(X,Y,2)
```

https://ww2.mathworks.cn/help/matlab/ref/trapz.html#buaijhw-1

# 目 录

- 数值微分
- 数值积分
- **微分方程解析解求解**
- 常微分方程数值求解
- 偏微分方程数值求解
- 课堂练习

# 微分方程解析解求解

- 对于通常的微分方程，可尝试采用**dsolve**函数求解其解析解：

```
S = dsolve(eqn)
S = dsolve(eqn,cond)
S = dsolve(___,Name,Value)
[y1,...,yN] = dsolve(___)
```

- eqn是一个符号方程，使用diff和==表示微分方程，如diff(y,x) == y表示方程dy/dx = y.
- cond为边界条件

- **示例：** 求解微分方程组

$$\frac{dy}{dt} = z$$

$$\frac{dz}{dt} = -y.$$

```
syms y(t) z(t)
eqns = [diff(y,t) == z, diff(z,t) == -y];
S = dsolve(eqns)
ySol(t) = S.y
zSol(t) = S.z
```

https://ww2.mathworks.cn/help/symbolic/dsolve.html

# 目 录

- 数值微分
- 数值积分
- 微分方程解析解求解
- **常微分方程数值求解**
- 偏微分方程数值求解
- 课堂练习

# ODE数值求解

- **常微分方程** (ordinary differential equation，**ODE**) 包含与一个自变量 t（通常称为时间）相关的因变量 y 的一个或多个导数。

- **ODE家族求解器：**

  对于无解析解的常微分方程（组），需要利用**ODE家族求解器**进行求。MATLAB中的常微分方程(ODE) 求解器可对具有各种属性的初始值问题进行求解。

$$\begin{pmatrix} y'_1 \\ y'_2 \\ \vdots \\ y'_n \end{pmatrix} = \begin{pmatrix} f_1(t, y_1, y_2, \ldots, y_n) \\ f_2(t, y_1, y_2, \ldots, y_n) \\ \vdots \\ f_n(t, y_1, y_2, \ldots, y_n) \end{pmatrix},$$

- **ODE方程组的写法示例：**

$$\begin{cases} y'_1 = y_2 \\ y'_2 = y_1 y_2 - 2. \end{cases}$$

```
function dy = myODE(t,y)
dy(1) = y(2);
dy(2) = y(1)*y(2)-2;
```

- **高阶ODE需转换成1阶ODE：**

$$\begin{aligned} y_1 &= y \\ y_2 &= y' \\ y_3 &= y'' \\ &\vdots \\ y_n &= y^{(n-1)}. \end{aligned}$$

# ODE数值求解

| 求解器 | 问题类型 | 精度 | 何时使用 |
|---|---|---|---|
| ode45 | 非刚性 | 中 | 大多数情况下，应当首先尝试求解器 ode45。 |
| ode23 | | 低 | 对于容差较宽松的问题或在刚度适中的情况下，ode23 可能比 ode45 更加高效。 |
| ode113 | | 低到高 | 对于具有严格误差容限的问题或在 ODE 函数需要大量计算开销的情况下，ode113 可能比 ode45 更加高效。 |
| ode15s | 刚性 | 低到中 | 若 ode45 失败或效率低下并且怀疑面临刚性问题，请尝试 ode15s。此外，当解算微分代数方程 (DAE) 时，请使用 ode15s。 |
| ode23s | | 低 | 对于误差容限较宽松的问题，ode23s 可能比 ode15s 更加高效。它可以解算一些刚性问题，而使用 ode15s 解算这些问题的效率不高。<br>ode23s 会在每一步计算 Jacobian，因此通过 odeset 提供 Jacobian 有利于最大限度地提高效率和精度。<br>如果存在质量矩阵，则它必须为常量矩阵。 |
| ode23t | | 低 | 对于仅仅是刚度适中的问题，并且需要没有数值阻尼的解，请使用 ode23t。 ode23t 可解算微分代数方程 (DAE)。 |
| ode23tb | | 低 | 与 ode23s 一样，对于误差容限较宽松的问题，ode23tb 求解器可能比 ode15s 更加高效。 |
| ode15i | 完全隐式 | 低 | 对于完全隐式问题 f(t,y,y’) = 0 和微分指数为 1 的微分代数方程 (DAE)，请使用 ode15i。 |

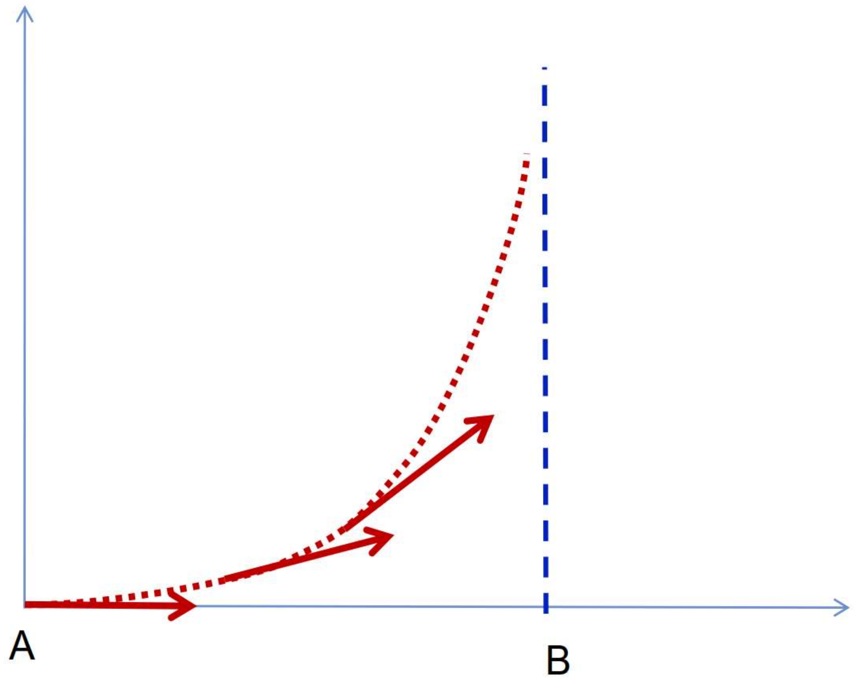
# ODE数值求解

- 对于ODE类问题，MATLAB提供了**一系列的求解器**
- ODE问题主要是非**刚性、刚性和完全隐性三种**
- 大部分情况下，OED问题是**非刚性问题，首选ode45进行求解。**但对于**精度要求更宽松或更严格的问题**而言，**ode23 和 ode113** 可能比 ode45 更加高效
- 一些 ODE 问题**具有较高的计算刚度或难度**。显式求解器（例如 ode45）获取解的速度慢得令人无法忍受。此时需要采用刚性求解器（ode15s等）
- 对于一个ODE问题，我们可以**首先尝试采用ode45等非刚性求解器**进行求解。当观察到非刚性求解器的**速度很慢**时，可尝试改用 **ode15s 等刚性求解器**

ODE求解器选择：
https://ww2.mathworks.cn/help/matlab/math/choose-an-ode-solver.html
关于非刚性ODE求解，可参见下列链接：
https://ww2.mathworks.cn/help/matlab/math/solve-nonstiff-odes.html
关于刚性ODE求解，可参见下列链接：
https://ww2.mathworks.cn/help/matlab/math/solve-stiff-odes.html

# ODE数值求解

**导弹追踪问题：**设位于坐标原点的甲舰向位于$x$轴上点A（1,0）处的乙舰发射导弹，导弹头始终对准乙舰。如果乙舰以最大速度$\upsilon_0$（常数）沿平行于$y$轴的直线行驶，导弹的速度为$5\upsilon_0$，求导弹的运行的曲线方程，以及乙舰行驶多远时，导弹将击中它?



# ODE数值求解

## 导弹追踪问题建模过程：

记导弹的速度为 $w$，乙舰的速率恒为 $v_0$，设时刻 t 乙舰的坐标为 $(X(t),Y(t))$，导弹的坐标为 $(x(t),y(t))$。当零时刻时，$(X(0),Y(0))=(1,0),(x(0),y(0))=(0,0)$ 建立微分方程模型：

$$\begin{cases} \dfrac{dx}{dt}=\dfrac{w}{\sqrt{(X-x)^2+(Y-y)^2}}(X-x) \\ \dfrac{dy}{dt}=\dfrac{w}{\sqrt{(X-x)^2+(Y-y)^2}}(Y-y) \end{cases}$$

因为乙舰以速度 $v_0$ 沿直线 $x=1$ 运动，设 $v_0=1$，$w=5$，$X=1$，$Y=t$，因此导弹运动轨迹的参数方程为

$$\begin{cases} \dfrac{dx}{dt}=\dfrac{5}{\sqrt{(1-x)^2+(t-y)^2}}(1-x) \\ \dfrac{dy}{dt}=\dfrac{5}{\sqrt{(1-x)^2+(t-y)^2}}(t-y) \\ x(0)=0,y(0)=0 \end{cases}$$

# ODE数值求解

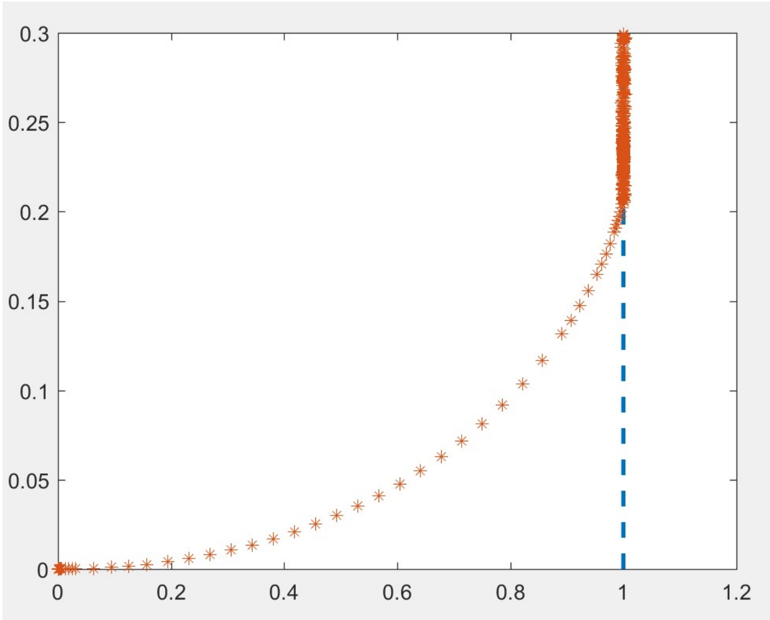
## 导弹追踪问题MATLAB编程求解过程：

➢ **方程组代码函数：**

```
function dy = eq2(t,y)
dy = zeros(2,1);
dy(1) = 5*(1-y(1))/sqrt((1-y(1))^2+(t-y(2))^2);
dy(2) = 5*(t-y(2))/sqrt((1-y(1))^2+(t-y(2))^2);
```

➢ **调用函数采用ode45求解器进行求解并绘制运动曲线：**

```
t0=0;  tf=0.3;
[t, y] = ode45('eq2', [t0 tf], [0 0]);
Y=t;  X(1:length(Y))=1;
figure;
plot(X,Y,'--','linewidth',2);   %乙舰的运动曲线
hold on
plot(y(:,1),y(:,2),'*');        %导弹的运动曲线
```




**从图中可以看出，在约0.2s时，导弹击中目标**

# 目 录

- 数值微分
- 数值积分
- 微分方程解析解求解
- 常微分方程数值求解
- **偏微分方程数值求解**
- 课堂练习

# PDE数值求解

- 当微分方程方程中不止一个变量时，称为**偏微分方程（PDE）**
- MATLAB偏微分方程的求解：**pdede函数和PDE工具箱**

| PDE方程（组）求解 | 特点 |
|---|---|
| pdede函数 | 具有较大的通用性，但只支持命令行调用方式 |
| PDE工具箱 | 能够求解一些常见的二阶PDE问题（特定的PDE问题），支持命令行调用（solvepde函数）和GUI界面操作两种方式。 |

当问题属于特定PDE范畴时，推荐使用PDE工具箱进行求解

# PDE数值求解

## ➢ pdede函数：

$$c\left(x,t,u,\frac{\partial u}{\partial x}\right)\frac{\partial u}{\partial t}=x^{-m}\frac{\partial}{\partial x}\left(x^{m}f\left(x,t,u,\frac{\partial u}{\partial x}\right)\right)+s\left(x,t,u,\frac{\partial u}{\partial x}\right).$$

当需要是用pdede求解偏微分方程（组）的时候，**需要首先将方程写成上式所示的形式**。上式中，PDE 适用于 $t_0 \le t \le t_f$ 且 $a \le x \le b$。[a, b] 必须是有限区间。m 可以是 0、1 或 2，分别对应平板、柱状或球面对称性。如果 $m > 0$，则 a 必须大于或等于 0。

关于pdede的详细用法请参见下列链接：

https://ww2.mathworks.cn/help/matlab/math/partial-differential-equations.html#bu8tnr1

# PDE数值求解

## ➤ PDE工具箱支持的方程类型：

$$m\frac{\partial^2 u}{\partial t^2}+d\frac{\partial u}{\partial t}-\nabla\cdot(c\nabla u)+au=f$$

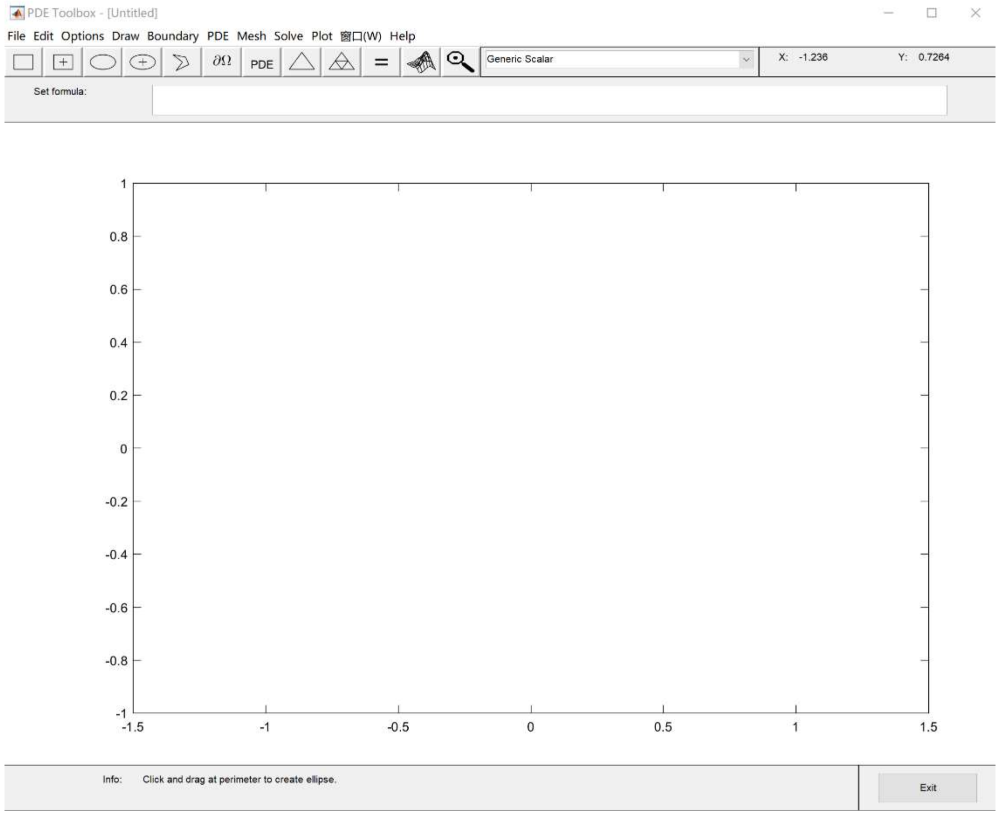
$$-\nabla\cdot(c\nabla u)+au=\lambda du$$

$$-\nabla\cdot(c\nabla u)+au=\lambda^2 mu$$

（MATLAB 2019b支持）

## ➤ PDE工具箱调用方法：

在MATLAB界面APP功能区里面选择（工具箱界面方式。APP功能区如果找不到，可能是没有安装，安装下PDE即可），也可以在命令行窗口输pdetool（2017a版本）或者pdeModeler（2019b）可以调用工具箱。



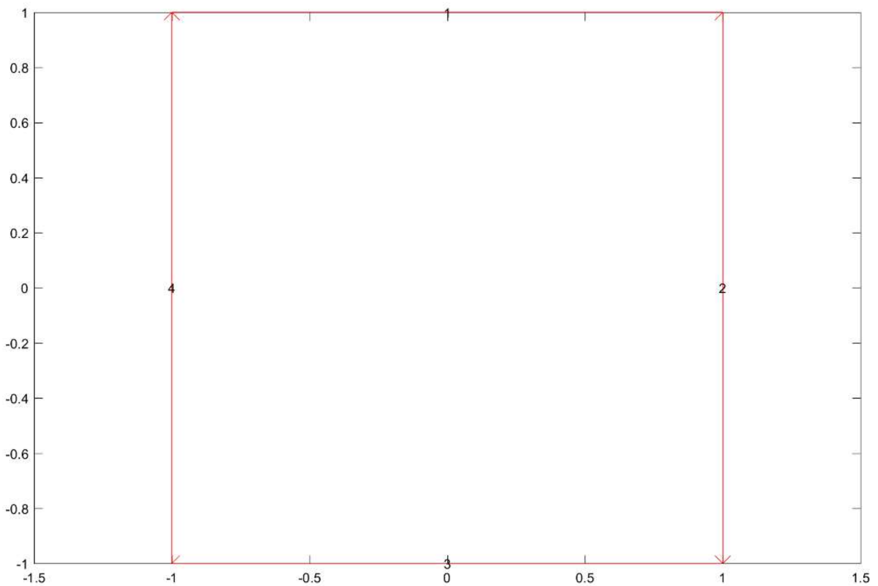
# PDE数值求解

## ➢ 案例：

考虑一个简单的二阶波动方程：

$$\frac{\partial^2 u}{\partial t^2} - \nabla \cdot \nabla \mathrm{u} = 0$$

对于上述波动方程求解下图边界1~4围成区域内t=30时u的分布。




边界和初始条件：

1和3处的边界条件为$\nabla \mathrm{u} = 0$

2和4处的边界条件为$\mathrm{u} = 0$

u(t=0) = atan[cos($\pi x/2$)]

u'(t=0)=3sin($\pi x$)$e^{\sin(\pi y/2)}$

# 目录


- 数值微分
- 数值积分
- 微分方程解析解求解
- 常微分方程数值求解
- 偏微分方程数值求解
- **课堂练习**

# 课堂练习

## ➢ 问题描述

**Lotka-Volterra方程是由两个一阶非线性ODE组成的方程组，用于描述生物系统中捕食者和猎物的种群。随着时间的推移，捕食者和猎物的数量会根据方程发生变化**

$$\frac{dx}{dt} = \alpha x - \beta xy,$$
$$\frac{dy}{dt} = \delta xy - \gamma y.$$

- **$x$是猎物的种群大小**
- **$y$是捕食者的种群大小**
- **$t$是时间**
- **$\alpha$、$\beta$、$\delta$和$\gamma$是描述两个物种之间交互的常量参数**

令$\alpha=\gamma=1$，$\beta=0.01$和$\delta=0.02$，初始种群大小为均为50，设置时间区间为[0,15]采用ode45求解器对方程组进行求解并绘制结果种群对时间的图。

# 谢谢！