北京航空航天大學
BEIHANG UNIVERSITY

2018 计算流体力学课程大作业

——2D 喷管流场模拟

姓　　名：　　王　勇

学　　号：　　14041113

任课教师：　　宁方飞

完成时间：　　2018 年　07 月　29 日

联系方式　　15810212023

1. 计算网格生成

在进行计算流体计算之前首先需要生成计算网格，即对计算域进行空间离散，计算域如图 1_1 所示，所计算内容为一带有远场的收-扩喷管流动。将计算区域划分为两个 Block，如图 1_2 所示，网格划分采用代数方法生成网格，并对网格进行适当加密处理。

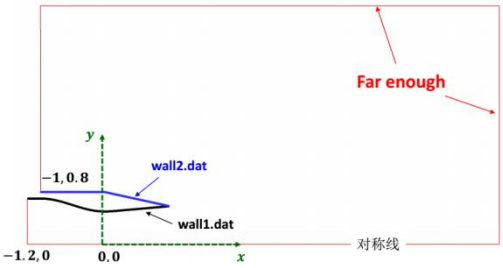


图 1_1 计算域

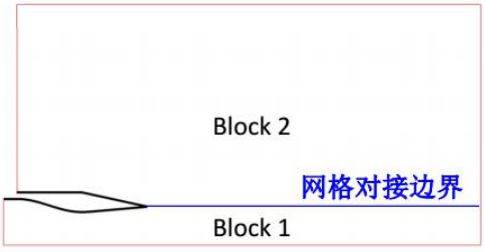


图 1_2 网格划分

计算远场轴向取至 x=6，y=2.8。对于 Block1：X 方向 331 个网格点，Y 方向 71 个网格点。对于 Block2：X 方向 261 个网格点，Y 方向 81 个网格点，进行结构化网格划分。对喉部区域，喷管出口区域，Block2 下部区域进行加密处理，采用指数方法进行加密，加密方式如式（1-1）、（1-2）所示，其中 $\alpha = 2$ 。其中喷管型面曲线部分采用五次方程进行拟合，拟合公式如式（1-3）所示：

$$\xi = x \tag{1-1}$$

$$\frac{e^{\alpha\eta} - 1}{e^{\alpha} - 1} = \frac{y - y_1(x)}{y_2(x) - y_1(x)} \tag{1-2}$$

$$\begin{aligned} y = {} & 1.79015e^{-2} * x^6 - 0.1761616 * x^5 - 0.5135756 * x^4 \\ & - 0.02836162 * x^3 + 0.4913228 * x^2 + 1.53865e^{-4} * x + 0.5000094 \end{aligned} \tag{1-3}$$

对与 Block1 的 y 方向，由 0 到 0.7 进行 70 等分。对于 Block2 的 y 方向，由 0.8 到 1.8 利用上式进行加密，共 60 个网格，由 1.8 到 2.8 进行 20 等分。

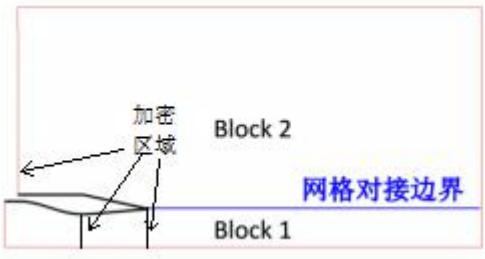


图 1_3 网格加密区域

将计算得到的数据节点坐标导入 tecplot 中进行网格图像显示如图 1_4 所示，网格总数为 43900。

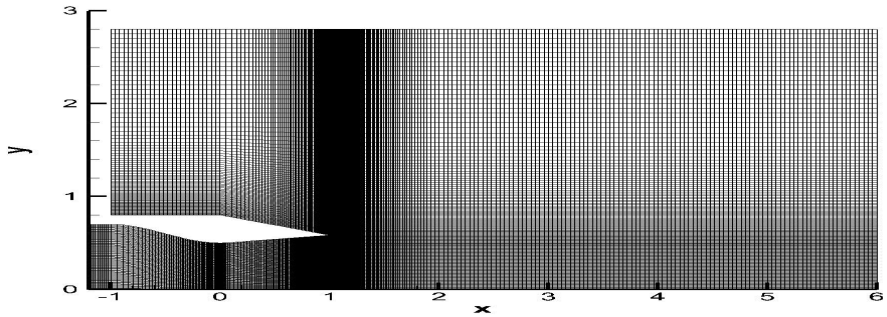


图 1_4 计算网格

2.计算格式

2.1 空间离散

2.1.1 对流迎风分裂格式介绍

在绕飞行器的流场中,尤其是高超声速飞行器的流场中经常存在激波、剪切流等流动参数急剧变化的区域,特别是在无粘流场中,这类区域的流动参数是不连续的。这给数值求解造成较大的困难。迄今为止,捕获流场中的激波、滑流面等间断现象的数值方法研究取得了重大的进展，并且发展了各种高分辨率的差分格式。

目前 CFD 常用的计算格式有中心格式、迎风格式以及总变差递减格式(Total Variation Diminishing Scheme, TVD)、保单调守恒格式(Monotonic Upwind Scheme for Conservation Laws, MUSCL)、本质无振荡格式(Essentially Non-Oscillatory Scheme, ENO)等。这些格式的应用使超声速、跨声速流场的计算有了很大的改进，从而可以模拟包含有激波、滑流面、粘性干扰、分离涡、真实气体效应等物理现象的复杂流场。其中，中心格式逻辑关系最简单、计算量小，但其分辨间断是通过人工粘性在一定程度上抹平间断而得到光滑解来实现的，它不仅数值耗散大，而且包含经验常数。TVD 格式构造复杂，计算量大，其构造机制可限制间断附近的振荡，但同时有较大的数值耗散，在非间断的极值附近也会导致格式降为一阶精度。

从 20 世纪 80 年代起，人们对迎风格式逐渐产生了浓厚的兴趣，随之进行了一系列的研究，并发现迎风格式对用 Euler 方程和 N-S 方程进行描述的广泛问题求解具有较高的计算效率和间断分辨率的综合优势，在一定限制条件下也可得到 TVD 的性质，因而受到广泛注意。到今天，迎风格式越来越成为数值计算领域主要的空间离散化技术之一。

自 1979 年 Steger 和 Warming 提出通量矢量分离(FVS)格式以来,人们相继提出各种迎风通量分裂格式，其中，Van Leer 提出的通量矢量分离(FVS)格式和 Roe 提出的通量差分分离(FDS)格式是最具有代表性、应用最为成功的两种迎风格式。FDS 格式采用矩阵计算对当地黎曼问题进行求解，具有高的间断分辨率，Roe、Osher 和 Godunov 曾经先后提出了不同的 FDS 格式，其中 Roe 的 FDS 因其突出的准确性和高效性而被普遍采用。但当其通量 Jacobian 矩阵的特征值很小时，会违反熵条件，产生非物理解，故必须引入熵修正，但熵修正引入了额外耗散，而且熵修正依赖于求解的问题，具有多种形式，包含经验常数，在高马赫数下 Roe 格式还会出现所谓的“Carbuncle(红宝石)”现象，即在强激波后产生非物理的紊乱信号。而 FVS 根据特征值的正负对通量进行分离，并且对守恒变量等标量进行插值计算避免了计算量的增加，是比 FDS 形式简单、计算效率高的计算格式。正因为这样，八十年代末到九十年代中期，大量的采用通量分裂的新格式被尝试用于 Euler 方程的数值计算，如：Van Leer 的 FVS 格式、对流迎风和压力分裂格式(Convective Upwind and Split Pressure, CUSP)、总焓不守恒的波/粒分裂格式(Wave/Particle split, WPS)以及 CUSP(WPS)格式和总焓守恒的水平对流迎风分裂格式(Advection Upstream Splitting Method, AUSM)等，并且取得了较好的计算效

果。但 Van Leer 的 FVS 格式还是存在数值耗散比较大，即便对于精确的接触间断条件，仍然存在数值通量，会抹平接触间断，从而导致的粘性区计算误差，而且通过简单的加密网格或是使用高阶差分并不能消除这种误差。这两种格式都追求高精度、高流场分辨率，而且希望消除数值振荡。然而,提高精度和分辨率常常与消除数值振荡相矛盾，为了消除振荡，传统的做法是在高精度格式中加入带调节系数的人工粘性。

2.1.2 Roe 格式求解对流数值通量$\widetilde{F}$算法

本文采用通量差分分裂 Roe 格式进行对流数值通量的求解计算，Roe 格式一般被称为通量差分分裂（简称 FDS）格式，由于其优秀的间断分辨率，成为目前应用最广、评价最高的 CFD 格式之一。前文以对 Roe 格式进行了介绍，此处不再赘述。

在进行计算流体力学计算时，即对已知积分形式的二维 Euler 方程组进行求解，方程组如下式所示：

$$\frac{\partial}{\partial t}\int_S \vec{Q}\cdot dS + \oint_{\partial S} \overline{\overline{F}}\cdot\vec{n}\cdot dl = 0 \tag{2-1}$$

$$\vec{Q}=\begin{bmatrix}\rho\\ \rho u\\ \rho\upsilon\\ E\end{bmatrix} \qquad \overline{\overline{F}}=\begin{bmatrix}\rho\vec{V}\\ \rho u\vec{V}+p\vec{i}\\ \rho\upsilon\vec{V}+p\vec{j}\\ \rho H\vec{V}\end{bmatrix}$$

其中

$$\vec{V}=u\cdot\vec{i}+\upsilon\cdot\vec{j}$$

补充状态方程

$$E=\frac{p}{\gamma-1}+\frac{1}{2}\rho\left(u^2+\upsilon^2\right),\ \gamma=1.4 \tag{2-2}$$

$$p=\rho RT$$

有限体积空间离散控制方程（2-3）：

$$\frac{\partial}{\partial t}\left(\vec{Q}\cdot S\right)_{i,j}+\widetilde{F}_{i+\frac{1}{2},j}-\widetilde{F}_{i-\frac{1}{2},j}+\widetilde{F}_{i,j+\frac{1}{2}}-\widetilde{F}_{i,j-\frac{1}{2}}=0 \tag{2-3}$$

采用 Roe 格式求解对流数值通量$\widetilde{F}$：

$$\tilde{F}\left(\bar{Q}_L,\bar{Q}_R,l\right)=\frac{1}{2}\left[\hat{F}\left(\bar{Q}_L,l\right)+\hat{F}\left(\bar{Q}_R,l\right)-\left|\tilde{A}\left(\bar{Q}_L,\bar{Q}_R,l\right)\right|\left(\bar{Q}_R-\bar{Q}_L\right)\right] \tag{2-4}$$

其中，l 为控制单元边线长度；下标 L，R 代表控制单元表面左、右两边的流场变量，可分别取 i-1 及 i（单元边线为 i-1 和 i 之间时）、j-1 及 j（单元边线为 j-1 和 j 之间时）单元中心处的流场变量值，此时格式为一阶精度，$\tilde{A}$ 为 Roe 平均矩阵；$\hat{F}\left(\bar{Q},l\right)$为广义坐标系下的对流通量

$$\hat{F}\left(\bar{Q},l\right)=l\begin{bmatrix}\rho U\\ \rho u U+n_x p\\ \rho \upsilon U+n_y p\\ \rho H U\end{bmatrix} \tag{2-5}$$

其中(n_x,n_y)为控制单元边线的法向单位矢量；对流速度 U 为$U=n_x u+n_y \upsilon$。总焓可表示为：

$$H=P/(\gamma-1)+\rho\left(u^2+v^2\right)/2 \tag{2-6}$$

式（2-4）中的$\left|\tilde{A}\right|\left(\bar{Q}_R-\bar{Q}_L\right)$项可以用下式直接计算

$$\left|\tilde{A}\right|\left(\bar{Q}_R-\bar{Q}_L\right)=\begin{bmatrix}\beta_4\\ \bar{u}\beta_4+n_x\beta_5+\beta_6\\ \bar{\upsilon}\beta_4+n_y\beta_5+\beta_7\\ \bar{H}\beta_4+\bar{U}\beta_5+\bar{u}\beta_6+\bar{\upsilon}\beta_7-c^2\beta_1/(\gamma-1)\end{bmatrix} \tag{2-7}$$

其中

$$\begin{aligned}
\beta_1 &= l\left|\lambda_1\right|\left(\Delta\rho-\Delta p/c^2\right)\\
\beta_2 &= l\left|\lambda_3\right|\left(\Delta p+\bar{\rho}c\Delta U\right)/\left(2c^2\right)\\
\beta_3 &= l\left|\lambda_2\right|\left(\Delta p-\bar{\rho}c\Delta U\right)/\left(2c^2\right)\\
\beta_4 &= \beta_1+\beta_2+\beta_3\\
\beta_5 &= c\left(\beta_2-\beta_3\right)\\
\beta_6 &= l\left|\lambda_1\right|\bar{\rho}\left(\Delta u-n_x\Delta U\right)\\
\beta_7 &= l\left|\lambda_1\right|\bar{\rho}\left(\Delta \upsilon-n_y\Delta U\right)
\end{aligned} \tag{2-8}$$

$\lambda_{1,2,3}$ 为 Roe 平均矩阵 $\tilde{A}$ 的特征值，它们分别定义为

$$\begin{aligned}\lambda_1 &= \overline{U} \\ \lambda_2 &= \overline{U} - c \\ \lambda_3 &= \overline{U} + c\end{aligned} \tag{2-9}$$

$\Delta(\cdot) \equiv (\cdot)_R - (\cdot)_L$，即控制单元表面左右（定义见前文）两边流场变量之差。式（2-7）和式（2-9）中的平均量$\overline{(\cdot)}$皆为 Roe 平均意义下的流场变量。Roe 平均定义为

$$\begin{cases}\overline{\rho} = \sqrt{\rho_L \rho_R} \\ \overline{q} = \dfrac{q_L\sqrt{\rho_L} + q_R\sqrt{\rho_R}}{\sqrt{\rho_L} + \sqrt{\rho_R}}, \qquad q = u, \upsilon, H, U\end{cases} \tag{2-10}$$

音速 c 定义为

$$c = \sqrt{(\gamma - 1)\left[\overline{H} - \tfrac{1}{2}\left(\overline{u}^2 + \overline{\upsilon}^2\right)\right]} \tag{2-11}$$

由于 Roe 格式是在线化 Riemann 问题的基础上构造的，在声速线或滞止点附近，Roe 平均矩阵的特征值$(\lambda_1、\lambda_2、\lambda_3)$趋于 0 时，格式中的内在的数值耗散将趋于消失，这时会出现伪解或计算不稳定。为防止这种情况发生，本文采用 Harten 的熵修正方法进行解决。

Harten 的熵修正方法：

$$|\lambda_i| = \begin{cases} |\lambda_i| & |\lambda_i| \geq \varepsilon \\ \dfrac{\lambda_i^2 + \varepsilon^2}{2\varepsilon} & |\lambda_i| < \varepsilon \end{cases} \tag{2-12}$$

ε 为一个小量，取值范围一般为 0.05～0.25，本文取为 0.1。

为获得更高阶精度的 Roe 格式，可以利用 MUSCL (Monotone Upstream-centered Scheme for Conservation Laws)插值构造单元界面左右两边的流动参数。

构造 MUSCL 差分格式的基本思想是：为了得到高阶精度的 Godunov 差分格式，差分格式从守恒型微分方程出发；其次，为了保证数值解是物理解，并且保证差分格式具有 r 阶精度，要求构造一个合适的重构函数$R(x, u^n)$，由它计算得到的数值通量$\tilde{F}_{j+1/2}$，在 MUSCL 构造过程中充分考虑了差分格式的单调性和

守恒性。

2.2 时间离散

对于时间离散，采用 1 阶单步法显示时间离散的方式，时间步长的 CFL 数取为 0.8。最大时间步长可以通过近似的方法获得。对于固定形状的网格，采用以下的表达式。

$$\Delta t_k = CFL \cdot \frac{\Omega_k}{\sum_{i=1}^{kedges} \left| u_i \Delta y_i - v_i \Delta x_i \right| + c_i \sqrt{\left(\Delta x_i^2 + \Delta y_i^2 \right)}} \tag{2-13}$$

则可得到

$$\vec{Q}^{n+1} = \vec{Q}^{n} - \Delta t_k \cdot \frac{\overline{\overline{F}}}{\Delta \Omega_i} \tag{2-14}$$

2. 边界条件与初场设置

2.1 进口边界

对于亚音进口条件，对于二维问题，给定三个边界条件，分别给定总温（$T_{in}^{*} = 330$）、总压（$P_{in}^{*} = 250000$）、气流进口角度（气流进口角度为 0，即 $v_{in} = 0$），选取静温赋予数值边条（即 $T_{in} = T_{inner}$）。

2.2 出口边界

对于出口边界条件，若出口为亚音速，则给定静压（$P_{out} = 85419$），其余变量赋予数值边条。若出口为超音速，则四个变量均赋予数值边条。

2.3 壁面边界

对于无粘流体，在物面边界上，应该满足流动方向与物面相切，及物面的法向速度分量为零，可知 $\vec{v} \cdot \vec{n} = 0$。这就是物面无穿透边界条件。而将压力与密度由内场值外推得到。

2.4 压力远场

采用一维 Riemann 不变量来处理远场边界条件。令 q_n, q_t 分别为边界外法向

速度和切向速度，根据特征线理论，Riemann 不变量可写成：

$$\begin{cases} R_1 = q_t \\ R_2 = s \\ R_3 = q_n - \dfrac{2a}{K-1} \\ R_4 = q_n + \dfrac{2a}{K-1} \end{cases}$$

对于本问题，由于远场为亚音速，Riemann 不变量的取值可分为以下两种情况：

1）亚音速入流（$-a < q_n < 0$）

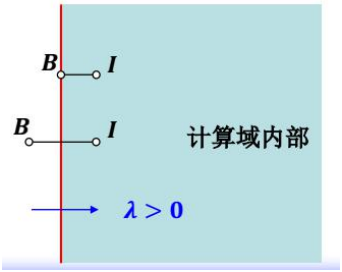
R_1，R_2, R_3 取来流值，R_4 取内场值。

2）亚音速出流（$0 < q_n < a$）

R_1，R_2, R_4 取内流值，R_3 取来流值。

2.5 交界面边界

在交界面上采用特征/无反射边界条件，设置虚网格进行计算，将 Block1 的交界面下层网格上各变量的值 I 赋予 Block2 上的虚网格 B 上，同理将 Block2 的交界面上层网格上各变量的值赋予 Block1 上边界上的虚网格上。



之后将虚网格作为计算边界进行迭代计算。

2.6 对称边界

在对称面上，在对称轴下方设置一层虚网格，该层虚网格上的变量值 ρ、p、u 均与对称轴上方一层网格数值相等，而速度 v 与该层网格数值相等，方向相反。

上述 4 个基本变量确定后，其余所有其他参数值均可由改 4 个变量确定。

不同的初场设置，对喷管流场的收敛性有一定影响。设置了三组不同初场进行对比，具体设置如下表所示。

表 2-1 初场参数值

初场编号	区域	轴向速度 u	径向速度 v	静压 p	静温 T
1	下部区域	0	0	101325	288.2
	上部区域	0	0	101325	288.2
2	下部区域	300	0	101325	288.2
	上部区域	150	0	85419	288.2
3	下部区域	400	0	101325	230
	上部区域	150	0	90000	288.2

不同初场的收敛曲线如下图所示，以能量方程为例，初场 1 迭代约 64550 步，方程的 dF 值保持不变，初场 2 需迭代约 62750 次，初场 3 需迭代约 62220 次，由此可见初场所给的值与最终计算得到的流场内各参数值越接近，计算收敛的速度也就越快。同时给定不同的初场，最终得到的流场是相同的。

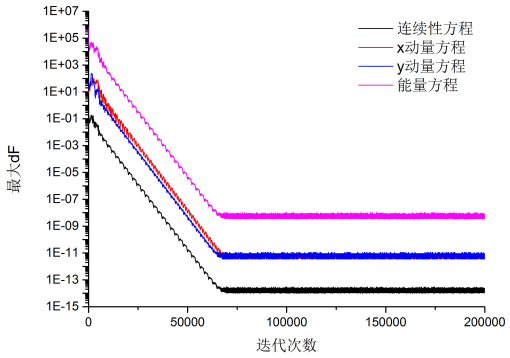


图 2_1 初场 1

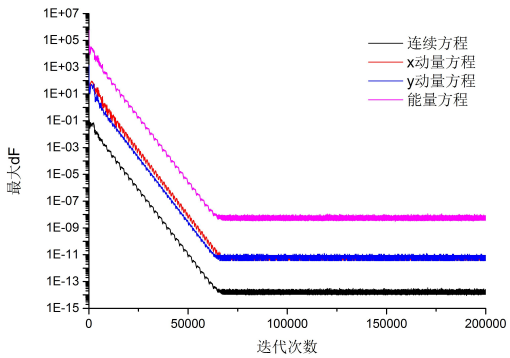


图 2_2 初场 2

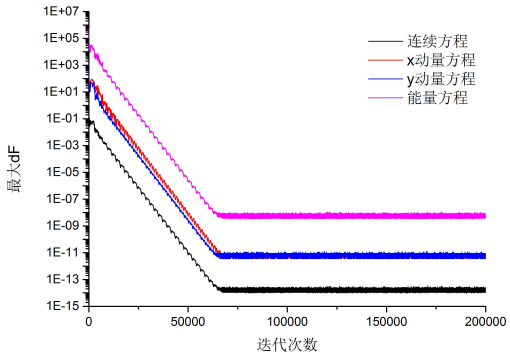


图 2_3 初场 3

3.程序求解流程

第一步：首先进行网格划分，分别对两个 Block 进行网格的划分，并利用两个三维数组对网格节点的坐标进行储存。

第二步：对网格节点上的各参数进行初始化处理，给定 0 速度初场，其与参数由远场总参数给定。之后进行网格矢量的计算，分别计算 i、j 方向的Δx与Δy以及边长。

第三步：根据第 3 节所述方法给定边界条件，对两个不同的 Block 分别设置边界条件。

第四步：利用 Roe 格式进行计算，首先求解对流速度 U，对于每一根控制单元的边线都需求解出边线两侧单元中心处流场变量所产生的对流速度。对于 I 方向即求边线的左右两侧对流速度，对于 J 方向即求边线上下两侧对流速度。

第五步：求解每个单元格相应的 H 焓值。

第六步：根据式（2-5）计算广义坐标下的对流通量$\hat{F}\left(\bar{Q},1\right)$，与求解对流速度相同，在求解对流通量的时候需要同时考虑每一根单元变现两侧单元中心处流场变量所产生的对流通量。对于 I 方向即求边线的左右两侧对流通量，对于 J 方向即求边线上下两侧对流速度。

第七步：根据式（5）求解式（3）中的$\left|\widetilde{A}\right|\left(\bar{Q}_R-\bar{Q}_L\right)$项，对于每一根边线仅需计算一次。公式中用到了边线两边单元中心处的流场变量值，通过求差和求 Roe 平均意义下的流场变量来计算。

第八步：根据求得的量，通过式（3）计算对流数值通量$\widetilde{F}$。计算方程中空间离散得到的$\widetilde{F}_{i+\frac{1}{2},j}-\widetilde{F}_{i-\frac{1}{2},j}+\widetilde{F}_{i,j+\frac{1}{2}}-\widetilde{F}_{i,j-\frac{1}{2}}$。

第九步：计算当前步的 Q 值，之后计算每个网格的当地时间步，再利用时间离散求解下一时间步的 Q 值，进一步得到各个变量。

第十步：迭代完成后，对数据进行后处理，并进行相应的输出。

4.计算结果分析

计算得到的残差曲线见第 2 节所示，由于不同初场得到的最终流场分布是相同的，选取初场 1 的计算结果进行分析。计算区域内的静温分布如图 4_1 所示，马赫数分布如图 4_2 所示。

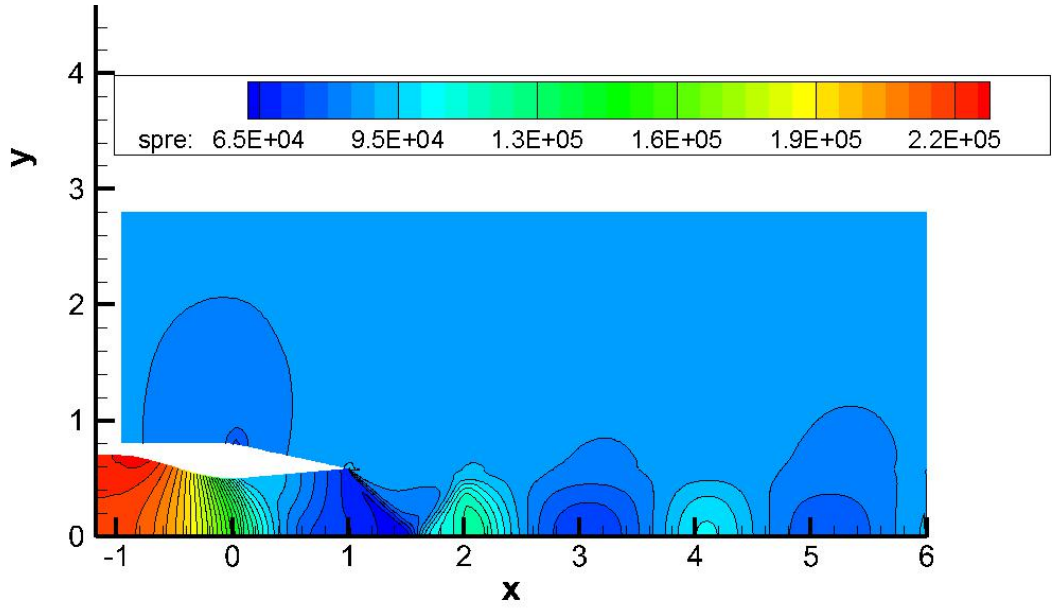


图 4_1 静压分布云图

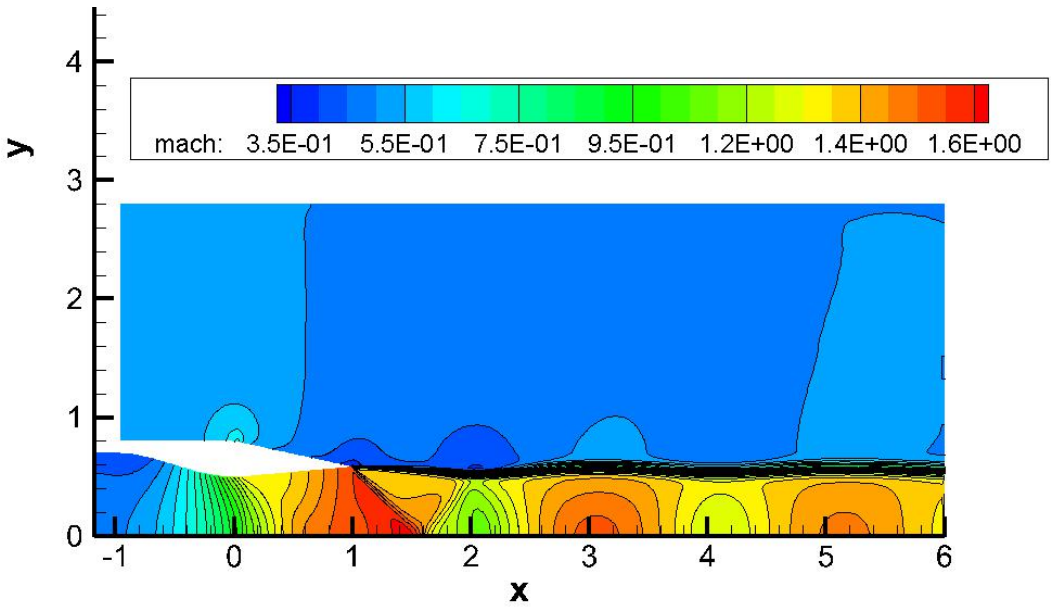


图 4_2 马赫数分布云图

根据计算结果可以看到，最终流场内的流动符合收-扩喷管的流动特性，气体在收敛段逐渐加速，静压逐渐减小，在到达喉道处气体达到声速，并在扩张段继续膨胀加速，静压继续减小。喷管出口处流场可由图 4_3 简化表示，在气体到达喷管出口 AO 处时，由于气体过膨胀，产生了斜激波 AB，气体在经过斜激波后马赫数降低，静压升高，随后由于远场的影响，产生了激波的反射和相交，在第一道斜激波与对称轴相交位置 B 发出一道膨胀波，该膨胀波打在自由边界上后又反射形成了一道斜激波 CD，由此不断循环。在自由边界处，喷流与远场两侧马赫数差距较大。气体在经过斜激波后在区域 ABC 静压升高至远场静压，此时由于区域 BCD 内静压较低，气流需经过一道膨胀波 BC 将气体静压降至区域 BCD 内的静压，此后不断循环进行，在向后的不断发展过程中，由于与远场的相互作用，斜激波即膨胀波均逐渐减弱。因此能够在压力云图及马赫数云图内看到一块块的高压区、低压区以及高马赫数区、低马赫数区。而对于远场位置，由于型面的影响，在经过第一段平直段后，进入一扩张段，马赫数有所降低，随后在与喷流的相互作用过程中，喷流的静压变化逐渐扩大到远场内。

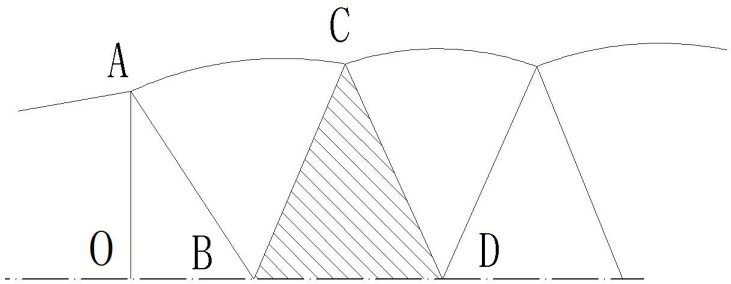


图 4_3 激波反射示意图

本文程序计算采用的是有限体积法空间离散，对流通量格式采用的是 Roe 格式，计算得到为一阶结果，因此精确度较 MUSCL 插值得到的结果稍差，同时由于计算过程中的虚网格设置问题，直接将虚网格上的数值作为了实际网格坐标节点上值进行计算，最终得到的计算结果存在一定的误差。本文计算的是无粘情况下喷管内的气体流动变化，因此壁面未采用无滑移条件，同时直接利用有限体积法离散二维 Euler 方程组进行求解，未考虑湍流耗散等的影响。但计算得到的流动已能基本反映喷管流场的变化。

5.程序源代码

```
#include <stdio.h>
#include <math.h>
#define e 2.718281828459
#define a 2
#define gama 1.4
#define R 287.06
#define cfl 0.8
	FILE *fp,*fq,*fr,*fs;
	double nodes[331][71][2],nodesc1[331][71][2],nodes2[261][81][2],nodesc2[261][81][2];
	  int i,j,k,m,maxi,maxj,max,maxi2,maxj2,max2;
	  double dyi_1[331][71],dxi_1[331][71],dyj_1[331][71],dxj_1[331][71];
	  double dyi_2[261][81],dxi_2[261][81],dyj_2[261][81],dxj_2[261][81];
	double sli_1[330][71],slj_1[331][71],area_1[330][71];
	double sli_2[261][81],slj_2[261][81],area_2[261][81];
	  double pho1_1[331][71],pre1_1[331][71],vx1_1[331][71],vy1_1[331][71],T1_1[331][71],ma1_1[331][71];
	  double pho2_1[261][81],pre2_1[261][81],vx2_1[261][81],vy2_1[261][81],T2_1[261][81],ma2_1[261][81];
	  double pho1_2[331][71],pre1_2[331][71],vx1_2[331][71],vy1_2[331][71],T1_2[331][71],ma1_2[331][71];
	double pho2_2[261][81],pre2_2[261][81],vx2_2[261][81],vy2_2[261][81],T2_2[261][81],ma2_2[261][81];
	double total_pre1_1[331][71],total_T1_1[331][71];
	double total_pre2_1[261][81],total_T2_1[261][81];
	double
Uir_1[331][71],Uil_1[331][71],Ujr_1[331][71],Ujl_1[331][71],H_1[331][71],Fjr_1[331][71][4],Fjl_1[331][71][4],Fir_1[331][71][4],Fil_
1[331][71][4];
	double
Uir_2[261][81],Uil_2[261][81],Ujr_2[261][81],Ujl_2[261][81],H_2[261][81],Fjr_2[261][81][4],Fjl_2[261][81][4],Fir_2[261][81][4],Fil_
2[261][81][4];
	double shengsu_1,pho_1_ba,vx1_1_ba,vy1_1_ba,H_1_ba,U_1_ba,lanmeta1_1,lanmeta2_1,lanmeta3_1;
	double shengsu_2,pho_2_ba,vx2_2_ba,vy2_2_ba,H_2_ba,U_2_ba,lanmeta1_2,lanmeta2_2,lanmeta3_2;
	double beta1_1,beta2_1,beta3_1,beta4_1,beta5_1,beta6_1,beta7_1;
	double beta1_2,beta2_2,beta3_2,beta4_2,beta5_2,beta6_2,beta7_2;
	double AQi_1[331][71][4],AQj_1[331][71][4],Flux_1[331][71][4],Q_1[331][71][4];
	double AQi_2[261][81][4],AQj_2[261][81][4],Flux_2[261][81][4],Q_2[261][81][4];
	double resm1,resave[4],imax,jmax,tyj,txj,tyi,txi,tsli,tslj,vi,vj,sonic,chvel,dt,tres1,real[331];
	double vnorm,vtemp,maxflux,maxflux2;

	  void mesh_generation()
	{
	  for(j=0;j<71;j++)
	  {
		for(i=0;i<11;i++)
		{
			nodes[i][j][0]=0.02*i-1.2;
			nodes[i][j][1]=0.01*j;
		}
		for(i=11;i<61;i++)
		{
			nodes[i][j][0]=(log((i-10)*(pow(e,a)-1)/50+1))/a-1;

	  nodes[i][j][1]=(0.0179015*pow(nodes[i][j][0],6)-0.1761616*pow(nodes[i][j][0],5)-0.5135756*pow(nodes[i][j][0],4)-0.02836162*
pow(nodes[i][j][0],3)+0.4913228*pow(nodes[i][j][0],2)+0.000153865*nodes[i][j][0]+0.5000094)/70*j;
		}
		for(i=61;i<71;i++)
		{
			nodes[131-i][j][0]=0.08983227479893685-0.08983227479893685*(log((i-61)*(pow(e,a)-1)/10+1))/a;
		}
		for(i=61;i<71;i++)
		{

	  nodes[i][j][1]=(0.0179015*pow(nodes[i][j][0],6)-0.1761616*pow(nodes[i][j][0],5)-0.5135756*pow(nodes[i][j][0],4)-0.02836162*
pow(nodes[i][j][0],3)+0.4913228*pow(nodes[i][j][0],2)+0.000153865*nodes[i][j][0]+0.5000094)/70*j;
		}
		for(i=71;i<151;i++)
		{
			nodes[i][j][0]=(1-0.08983227479893685)*(log((i-70)*(pow(e,a)-1)/80+1))/a+0.08983227479893685;
			nodes[i][j][1]=(0.08748864*nodes[i][j][0]+0.4960966)/70*j;
		}
		for(i=151;i<231;i++)
		{
			nodes[381-i][j][0]=2-(log((i-151)*(pow(e,a)-1)/80+1))/a;
			nodes[i][j][1]=0.583585301908299/70*j;
		}
		for(i=231;i<331;i++)
		{
			nodes[i][j][0]=0.04*(i-230)+2;
			nodes[i][j][1]=0.583585301908299/70*j;
```

```
            }
        }
        //上半区
            for(j=0;j<61;j++)                       //加密区
        {
            for(i=0;i<26;i++)
            {
                nodes2[i][j][0]=0.04*i-1;
                nodes2[i][60-j][1]=1.8-(log(j*(pow(e,a)-1)/60+1))/a;
            }
            for(i=26;i<81;i++)
            {
                nodes2[i][j][0]=(log((i-25)*(pow(e,a)-1)/55+1))/a;
                nodes2[i][60-j][1]=1.8-(1.8-(-0.2164147*nodes2[i][j][0]+0.8))*(log(j*(pow(e,a)-1)/60+1))/a;
            }
            for(i=81;i<161;i++)
            {
                nodes2[241-i][j][0]=2-(log((i-81)*(pow(e,a)-1)/80+1))/a;
                nodes2[i][60-j][1]=1.8-(1.8-0.583585301908299)*(log(j*(pow(e,a)-1)/60+1))/a;
            }
            for(i=161;i<261;i++)
            {
                nodes2[i][j][0]=0.04*(i-160)+2;
                nodes2[i][60-j][1]=1.8-(1.8-0.583585301908299)*(log(j*(pow(e,a)-1)/60+1))/a;
            }
        }
        for(j=61;j<81;j++)                  //非加密区
        {
            for(i=0;i<26;i++)
            {
                nodes2[i][j][0]=0.04*i-1;
                nodes2[i][j][1]=1.8+0.05*(j-60);
            }
            for(i=26;i<81;i++)
            {
                nodes2[i][j][0]=(log((i-25)*(pow(e,a)-1)/55+1))/a;
                nodes2[i][j][1]=1.8+0.05*(j-60);
            }
            for(i=81;i<161;i++)
            {
                nodes2[241-i][j][0]=2-(log((i-81)*(pow(e,a)-1)/80+1))/a;
                nodes2[i][j][1]=1.8+0.05*(j-60);
            }
            for(i=161;i<261;i++)
            {
                nodes2[i][j][0]=0.04*(i-160)+2;
                nodes2[i][j][1]=1.8+0.05*(j-60);
            }
        }
}
    void initialize()           //初始化
    {
          for(j=0;j<71;j++)
    {
      for(i=0;i<331;i++)
      {
            vx1_1[i][j]=0;
            vy1_1[i][j]=0;
            pre1_1[i][j]=101325;
            T1_1[i][j]=288.2;
            pho1_1[i][j]=pre1_1[i][j]/T1_1[i][j]/R;
            ma1_1[i][j]=sqrt(vx1_1[i][j]*vx1_1[i][j]+vy1_1[i][j]*vy1_1[i][j])/sqrt(gama*R*T1_1[i][j]);
            }
      }
      for(j=0;j<81;j++)
    {
      for(i=0;i<261;i++)
      {
            vx2_1[i][j]=0;
            vy2_1[i][j]=0;
            pre2_1[i][j]=101325;
            T2_1[i][j]=288.2;
            pho2_1[i][j]=pre2_1[i][j]/T2_1[i][j]/R;
            ma2_1[i][j]=sqrt(vx2_1[i][j]*vx2_1[i][j]+vy2_1[i][j]*vy2_1[i][j])/sqrt(gama*R*T2_1[i][j]);
            }
      }
}
    void area_caculate()            //计算网格矢量
    {
      for(j=0;j<71;j++)
```

```
{
  for(i=0;i<330;i++)
  {
        dyi_1[i][j]=nodes[i][j][1]-nodes[i+1][j][1];
        dxi_1[i][j]=nodes[i+1][j][0]-nodes[i][j][0];
        sli_1[i][j]=sqrt(dyi_1[i][j]*dyi_1[i][j]+dxi_1[i][j]*dxi_1[i][j]);
        }
  }
  for(j=0;j<70;j++)
{
  for(i=0;i<331;i++)
  {
        dyj_1[i][j]=nodes[i][j+1][1]-nodes[i][j][1];
        dxj_1[i][j]=nodes[i][j][0]-nodes[i][j+1][0];
        slj_1[i][j]=sqrt(dyj_1[i][j]*dyj_1[i][j]+dxj_1[i][j]*dxj_1[i][j]);
        }
  }
  for(j=0;j<70;j++)
  {
        for(i=0;i<330;i++)
        {
              area_1[i][j]=(dyj_1[i][j]+dyj_1[i+1][j])*dxi_1[i][j]/2;
        }
  }
          for(j=0;j<81;j++)
{
  for(i=0;i<260;i++)
  {
        dyi_2[i][j]=nodes2[i][j][1]-nodes2[i+1][j][1];
        dxi_2[i][j]=nodes2[i+1][j][0]-nodes2[i][j][0];
        sli_2[i][j]=sqrt(dyi_2[i][j]*dyi_2[i][j]+dxi_2[i][j]*dxi_2[i][j]);
        }
  }
  for(j=0;j<80;j++)
{
  for(i=0;i<261;i++)
  {
        dyj_2[i][j]=nodes2[i][j+1][1]-nodes2[i][j][1];
        dxj_2[i][j]=nodes2[i][j][0]-nodes2[i][j+1][0];
        slj_2[i][j]=sqrt(dyj_2[i][j]*dyj_2[i][j]+dxj_2[i][j]*dxj_2[i][j]);
        }
  }
  for(j=0;j<80;j++)
  {
        for(i=0;i<260;i++)
        {
              area_2[i][j]=(dyj_2[i][j]+dyj_2[i+1][j])*dxi_2[i][j]/2;
        }
  }
  }
double tao(double Ma)          //定义气动函数
{
  double taoma;
  taoma=1/(1+(gama-1)*Ma*Ma/2);
  return taoma;
  }
  double pai(double Ma)
  {
        double paima;
        paima=pow(1/(1+(gama-1)*Ma*Ma/2),gama/(gama-1));
        return paima;
  }
  void boundary_conditions()
{
  for(j=1;j<70;j++)              //进口边界条件 分区 1 亚音进口，一个变量外推
  {
      total_pre1_1[0][j]=250000;
      total_T1_1[0][j]=330;
      vy1_1[0][j]=0;
      if(T1_1[1][j]>330)
      {
        T1_1[0][j]=330;
              }
              else
              {
                    T1_1[0][j]=T1_1[1][j];
              }                                 //静温外推
      ma1_1[0][j]=sqrt((total_T1_1[0][j]/T1_1[0][j]-1)*2/(gama-1));
      pre1_1[0][j]=total_pre1_1[0][j]*pai(ma1_1[0][j]);
      pho1_1[0][j]=pre1_1[0][j]/R/T1_1[0][j];
```

```
    vx1_1[0][j]=ma1_1[0][j]*sqrt(gama*R*T1_1[0][j]);
      }
      for(j=1;j<70;j++)                    //出口边界条件
      {
            if(ma1_1[329][j]>=1)            //超音全部外推
            {
                  vx1_1[330][j]=vx1_1[329][j];
                  vy1_1[330][j]=vy1_1[329][j];
                  pre1_1[330][j]=pre1_1[329][j];
                  T1_1[330][j]=T1_1[329][j];
                  pho1_1[330][j]=pre1_1[330][j]/R/T1_1[330][j];

ma1_1[330][j]=sqrt(vx1_1[330][j]*vx1_1[330][j]+vy1_1[330][j]*vy1_1[330][j])/sqrt(gama*R*T1_1[330][j]);
            }
            else                        //亚音 3 个外推     静压给定
            {
                  pre1_1[330][j]=85419;
                  vx1_1[330][j]=vx1_1[329][j];
                  vy1_1[330][j]=vy1_1[329][j];
                  T1_1[330][j]=T1_1[329][j];
                  pho1_1[330][j]=pre1_1[330][j]/R/T1_1[330][j];

ma1_1[330][j]=sqrt(vx1_1[330][j]*vx1_1[330][j]+vy1_1[330][j]*vy1_1[330][j])/sqrt(gama*R*T1_1[330][j]);
            }
      }
      for(i=1;i<151;i++)                  //喷管壁面边界条件
      {
          vnorm=sqrt(vy1_1[i][69]*vy1_1[i][69]+vx1_1[i][69]*vx1_1[i][69]);
          vtemp=-(dyi_1[i][70])/dxi_1[i][70];
          vtemp=atan(vtemp);
            pho1_1[i][70]=pho1_1[i][69];
            vx1_1[i][70]=-vx1_1[i][69]+2*cos(vtemp)*vnorm;
            vy1_1[i][70]=-vy1_1[i][69]+2*sin(vtemp)*vnorm;
            pre1_1[i][70]=pre1_1[i][69];
            T1_1[i][70]=pre1_1[i][70]/pho1_1[i][70]/R;
            ma1_1[i][70]=sqrt(vx1_1[i][70]*vx1_1[i][70]+vy1_1[i][70]*vy1_1[i][70])/sqrt(gama*R*T1_1[i][70]);
      }
      for(i=1;i<330;i++)              //对称边界条件
      {
            pho1_1[i][0]=pho1_1[i][1];
            vx1_1[i][0]=vx1_1[i][1];
            vy1_1[i][0]=0;
            pre1_1[i][0]=pre1_1[i][1];
            T1_1[i][0]=pre1_1[i][0]/pho1_1[i][0]/R;
            ma1_1[i][0]=sqrt(vx1_1[i][0]*vx1_1[i][0]+vy1_1[i][0]*vy1_1[i][0])/sqrt(gama*R*T1_1[i][0]);
       }
       for(j=1;j<80;j++)              //进口边界条件  分区 1  亚音进口，一个变量外推
{

    total_pre2_1[0][j]=101325;
    total_T2_1[0][j]=288.2;
    ma2_1[0][j]=0.5;
    //vy1_1[0][j]=0;
    pre2_1[0][j]=total_pre2_1[0][j]*pai(ma2_1[0][j]);
    T2_1[0][j]=total_T2_1[0][j]*tao(ma2_1[0][j]);
    pho2_1[0][j]=pre2_1[0][j]/T2_1[0][j]/R;
    vy2_1[0][j]=vy2_1[1][j];
    vx2_1[0][j]=sqrt(ma2_1[0][j]*sqrt(gama*R*T2_1[0][j])*ma2_1[0][j]*sqrt(gama*R*T2_1[0][j])-vy2_1[0][j]*vy2_1[0][j]);
      }
      for(j=1;j<80;j++)     //////出口边界
      {
            if(ma2_1[259][j]>=1)            //超音全部外推
            {
                  vx2_1[260][j]=vx2_1[259][j];
                  vy2_1[260][j]=vy2_1[259][j];
                  pre2_1[260][j]=pre2_1[259][j];
                  T2_1[260][j]=T2_1[259][j];
                  pho2_1[260][j]=pre2_1[260][j]/R/T2_1[260][j];

ma2_1[260][j]=sqrt(vx2_1[260][j]*vx2_1[260][j]+vy2_1[260][j]*vy2_1[260][j])/sqrt(gama*R*T2_1[260][j]);
            }
            else                        //亚音 3 个外推     静压给定
            {
                  pre2_1[260][j]=85419;
                  vx2_1[260][j]=vx2_1[259][j];
                  vy2_1[260][j]=vy2_1[259][j];
                  T2_1[260][j]=T2_1[259][j];
                  pho2_1[260][j]=pre2_1[260][j]/R/T2_1[260][j];
```

```
ma2_1[260][j]=sqrt(vx2_1[260][j]*vx2_1[260][j]+vy2_1[260][j]*vy2_1[260][j])/sqrt(gama*R*T2_1[260][j]);
            }
        }
        for(i=1;i<260;i++)              //喷管壁面边界条件
        {
        if(vy2_1[i][79]>=0)       //出口，给一推三静压
            {
                pre2_1[i][80]=85419;
                vx2_1[i][80]=vx2_1[i][79];
                vy2_1[i][80]=vy2_1[i][79];
                T2_1[i][80]=T2_1[i][79];
                pho2_1[i][80]=pre2_1[i][80]/R/T2_1[i][80];
                ma2_1[i][80]=sqrt(vx2_1[i][80]*vx2_1[i][80]+vy2_1[i][80]*vy2_1[i][80])/sqrt(gama*R*T2_1[i][80]);
            }
            else                    //进口 ，给三推一
            {
                pre2_1[i][80]=85419;
              ma2_1[i][80]=0.5;
                T2_1[i][80]=288.2*tao(ma2_1[i][80]);
                vy2_1[i][80]=vy2_1[i][79];
                pho2_1[i][80]=pre2_1[i][80]/R/T2_1[i][80];

vx2_1[i][80]=sqrt(ma2_1[i][80]*sqrt(gama*R*T2_1[i][80])*ma2_1[i][80]*sqrt(gama*R*T2_1[i][80])-vy2_1[i][80]*vy2_1[i][80])
;
            }
        }
        for(i=1;i<81;i++)             //下壁面条件
        {
            vnorm=sqrt(vy2_1[i][1]*vy2_1[i][1]+vx2_1[i][1]*vx2_1[i][1]);
          vtemp=-(dyi_2[i][0])/dxi_2[i][0];
          vtemp=atan(vtemp);
            pho2_1[i][0]=pho2_1[i][1];
            vx2_1[i][0]=-vx2_1[i][1]+2*cos(vtemp)*vnorm;
            vy2_1[i][0]=-vy2_1[i][1]+2*sin(vtemp)*vnorm;
            pre2_1[i][0]=pre2_1[i][1];
            T2_1[i][0]=pre2_1[i][0]/pho2_1[i][0]/R;
            ma2_1[i][0]=sqrt(vx2_1[i][0]*vx2_1[i][0]+vy2_1[i][0]*vy2_1[i][0])/sqrt(gama*R*T2_1[i][0]);
         }


          for(i=151;i<330;i++)
        {
            pho1_1[i][70]=pho2_1[i-70][1];
            pre1_1[i][70]=pre2_1[i-70][1];
            vx1_1[i][70]=vx2_1[i-70][1];
            vy1_1[i][70]=vy2_1[i-70][1];
            T1_1[i][70]=pre1_1[i][70]/pho1_1[i][70]/R;
            ma1_1[i][70]=sqrt(vx1_1[i][70]*vx1_1[i][70]+vy1_1[i][70]*vy1_1[i][70])/sqrt(gama*R*T1_1[i][70]);

            pho2_1[i-70][0]=pho1_1[i][69];
            pre2_1[i-70][0]=pre1_1[i][69];
            vx2_1[i-70][0]=vx1_1[i][69];
            vy2_1[i-70][0]=vy1_1[i][69];
            T2_1[i-70][0]=pre2_1[i-70][0]/pho2_1[i-70][0]/R;
            ma2_1[i-70][0]=sqrt(vx2_1[i-70][0]*vx2_1[i-70][0]+vy2_1[i-70][0]*vy2_1[i-70][0])/sqrt(gama*R*T2_1[i-70][0]);
        }
}
void roe()                          //利用 roe 格式求解
{
        for(j=1;j<70;j++)
        {
            for(i=1;i<331;i++)
            {
                Uir_1[i][j]=dyj_1[i][j]/slj_1[i][j]*vx1_1[i][j]+dxj_1[i][j]/slj_1[i][j]*vy1_1[i][j];   //沿流向流动通过边界的
速度
              Uil_1[i][j]=dyj_1[i][j]/slj_1[i][j]*vx1_1[i-1][j]+dxj_1[i][j]/slj_1[i][j]*vy1_1[i-1][j];
            }
        }
        for(j=1;j<71;j++)
        {
            for(i=1;i<330;i++)
            {
                Ujr_1[i][j]=dyi_1[i][j]/sli_1[i][j]*vx1_1[i][j]+dxi_1[i][j]/sli_1[i][j]*vy1_1[i][j];   //沿垂直流向流动通过边
界的速度
              Ujl_1[i][j]=dyi_1[i][j]/sli_1[i][j]*vx1_1[i][j-1]+dxi_1[i][j]/sli_1[i][j]*vy1_1[i][j-1];
            }
        }
        for(j=0;j<71;j++)
```

```
        {
            for(i=0;i<331;i++)
            {

H_1[i][j]=(pre1_1[i][j]/(gama-1)+0.5*pho1_1[i][j]*(pow(vy1_1[i][j],2)+pow(vx1_1[i][j],2))+pre1_1[i][j])/pho1_1[i][j];
            }
        }
        for(j=1;j<80;j++)
        {
            for(i=1;i<261;i++)
            {
                Uir_2[i][j]=dyj_2[i][j]/slj_2[i][j]*vx2_1[i][j]+dxj_2[i][j]/slj_2[i][j]*vy2_1[i][j];   //沿流向流动通过边界的
速度
                Uil_2[i][j]=dyj_2[i][j]/slj_2[i][j]*vx2_1[i-1][j]+dxj_2[i][j]/slj_2[i][j]*vy2_1[i-1][j];
            }
        }
        for(j=1;j<81;j++)
        {
            for(i=1;i<260;i++)
            {
                Ujr_2[i][j]=dyi_2[i][j]/sli_2[i][j]*vx2_1[i][j]+dxi_2[i][j]/sli_2[i][j]*vy2_1[i][j];   //沿垂直流向流动通过边
界的速度
                Ujl_2[i][j]=dyi_2[i][j]/sli_2[i][j]*vx2_1[i][j-1]+dxi_2[i][j]/sli_2[i][j]*vy2_1[i][j-1];
            }
        }
        for(j=0;j<81;j++)
        {
            for(i=0;i<261;i++)
            {

H_2[i][j]=(pre2_1[i][j]/(gama-1)+0.5*pho2_1[i][j]*(pow(vy2_1[i][j],2)+pow(vx2_1[i][j],2))+pre2_1[i][j])/pho2_1[i][j];
            }
        }
        //I 方向的 F 求解
        for(j=1;j<70;j++)
        {
            for(i=1;i<331;i++)
            {
                Fil_1[i][j][0]=slj_1[i][j]*pho1_1[i-1][j]*Uil_1[i][j];                  //左侧对流通量
                Fil_1[i][j][1]=slj_1[i][j]*pho1_1[i-1][j]*vx1_1[i-1][j]*Uil_1[i][j]+dyj_1[i][j]*pre1_1[i-1][j];
                Fil_1[i][j][2]=slj_1[i][j]*pho1_1[i-1][j]*vy1_1[i-1][j]*Uil_1[i][j]+dxj_1[i][j]*pre1_1[i-1][j];
                Fil_1[i][j][3]=slj_1[i][j]*pho1_1[i-1][j]*H_1[i-1][j]*Uil_1[i][j];

                Fir_1[i][j][0]=slj_1[i][j]*pho1_1[i][j]*Uir_1[i][j];                  //右侧对流通量
                Fir_1[i][j][1]=slj_1[i][j]*pho1_1[i][j]*vx1_1[i][j]*Uir_1[i][j]+dyj_1[i][j]*pre1_1[i][j];
                Fir_1[i][j][2]=slj_1[i][j]*pho1_1[i][j]*vy1_1[i][j]*Uir_1[i][j]+dxj_1[i][j]*pre1_1[i][j];
                Fir_1[i][j][3]=slj_1[i][j]*pho1_1[i][j]*H_1[i][j]*Uir_1[i][j];
            }
        }
     for(j=1;j<80;j++)
        {
            for(i=1;i<261;i++)
            {
                Fil_2[i][j][0]=slj_2[i][j]*pho2_1[i-1][j]*Uil_2[i][j];                  //左侧对流通量
                Fil_2[i][j][1]=slj_2[i][j]*pho2_1[i-1][j]*vx2_1[i-1][j]*Uil_2[i][j]+dyj_2[i][j]*pre2_1[i-1][j];
                Fil_2[i][j][2]=slj_2[i][j]*pho2_1[i-1][j]*vy2_1[i-1][j]*Uil_2[i][j]+dxj_2[i][j]*pre2_1[i-1][j];
                Fil_2[i][j][3]=slj_2[i][j]*pho2_1[i-1][j]*H_2[i-1][j]*Uil_2[i][j];

                Fir_2[i][j][0]=slj_2[i][j]*pho2_1[i][j]*Uir_2[i][j];                  //右侧对流通量
                Fir_2[i][j][1]=slj_2[i][j]*pho2_1[i][j]*vx2_1[i][j]*Uir_2[i][j]+dyj_2[i][j]*pre2_1[i][j];
                Fir_2[i][j][2]=slj_2[i][j]*pho2_1[i][j]*vy2_1[i][j]*Uir_2[i][j]+dxj_2[i][j]*pre2_1[i][j];
                Fir_2[i][j][3]=slj_2[i][j]*pho2_1[i][j]*H_2[i][j]*Uir_2[i][j];
            }
        }
        //J 方向的 F 求解
        for(j=1;j<71;j++)
        {
            for(i=1;i<330;i++)
            {
                Fjl_1[i][j][0]=sli_1[i][j]*pho1_1[i][j-1]*Ujl_1[i][j];                  //左侧通量
                Fjl_1[i][j][1]=sli_1[i][j]*pho1_1[i][j-1]*vx1_1[i][j-1]*Ujl_1[i][j]+dyi_1[i][j]*pre1_1[i][j-1];
                Fjl_1[i][j][2]=sli_1[i][j]*pho1_1[i][j-1]*vy1_1[i][j-1]*Ujl_1[i][j]+dxi_1[i][j]*pre1_1[i][j-1];
                Fjl_1[i][j][3]=sli_1[i][j]*pho1_1[i][j-1]*H_1[i][j-1]*Ujl_1[i][j];

                Fjr_1[i][j][0]=sli_1[i][j]*pho1_1[i][j]*Ujr_1[i][j];                  //右侧通量
                Fjr_1[i][j][1]=sli_1[i][j]*pho1_1[i][j]*vx1_1[i][j]*Ujr_1[i][j]+dyi_1[i][j]*pre1_1[i][j];
                Fjr_1[i][j][2]=sli_1[i][j]*pho1_1[i][j]*vy1_1[i][j]*Ujr_1[i][j]+dxi_1[i][j]*pre1_1[i][j];
                Fjr_1[i][j][3]=sli_1[i][j]*pho1_1[i][j]*H_1[i][j]*Ujr_1[i][j];
            }
```

```
        }
          for(j=1;j<81;j++)
        {
            for(i=1;i<260;i++)
            {
                Fjl_2[i][j][0]=sli_2[i][j]*pho2_1[i][j-1]*Ujl_2[i][j];                    //左侧通量
                Fjl_2[i][j][1]=sli_2[i][j]*pho2_1[i][j-1]*vx2_1[i][j-1]*Ujl_2[i][j]+dyi_2[i][j]*pre2_1[i][j-1];
                Fjl_2[i][j][2]=sli_2[i][j]*pho2_1[i][j-1]*vy2_1[i][j-1]*Ujl_2[i][j]+dxi_2[i][j]*pre2_1[i][j-1];
                Fjl_2[i][j][3]=sli_2[i][j]*pho2_1[i][j-1]*H_2[i][j-1]*Ujl_2[i][j];

                Fjr_2[i][j][0]=sli_2[i][j]*pho2_1[i][j]*Ujr_2[i][j];                    //右侧通量
                Fjr_2[i][j][1]=sli_2[i][j]*pho2_1[i][j]*vx2_1[i][j]*Ujr_2[i][j]+dyi_2[i][j]*pre2_1[i][j];
                Fjr_2[i][j][2]=sli_2[i][j]*pho2_1[i][j]*vy2_1[i][j]*Ujr_2[i][j]+dxi_2[i][j]*pre2_1[i][j];
                Fjr_2[i][j][3]=sli_2[i][j]*pho2_1[i][j]*H_2[i][j]*Ujr_2[i][j];
            }
        }
        for(j=1;j<70;j++)
        {
            for(i=1;i<331;i++)
            {
                pho_1_ba=sqrt(pho1_1[i-1][j]*pho1_1[i][j]);                    //Roe 平均定义

    vx1_1_ba=(vx1_1[i-1][j]*sqrt(pho1_1[i-1][j])+vx1_1[i][j]*sqrt(pho1_1[i][j]))/(sqrt(pho1_1[i-1][j])+sqrt(pho1_1[i][j]));

    vy1_1_ba=(vy1_1[i-1][j]*sqrt(pho1_1[i-1][j])+vy1_1[i][j]*sqrt(pho1_1[i][j]))/(sqrt(pho1_1[i-1][j])+sqrt(pho1_1[i][j]));

    H_1_ba=(H_1[i-1][j]*sqrt(pho1_1[i-1][j])+H_1[i][j]*sqrt(pho1_1[i][j]))/(sqrt(pho1_1[i-1][j])+sqrt(pho1_1[i][j]));

    U_1_ba=(Uil_1[i][j]*sqrt(pho1_1[i-1][j])+Uir_1[i][j]*sqrt(pho1_1[i][j]))/(sqrt(pho1_1[i-1][j])+sqrt(pho1_1[i][j]));
                shengsu_1=sqrt((gama-1)*(H_1_ba-(vx1_1_ba*vx1_1_ba+vy1_1_ba*vy1_1_ba)/2));
                lanmeta1_1=U_1_ba;
                lanmeta2_1=U_1_ba-shengsu_1;
                lanmeta3_1=U_1_ba+shengsu_1;
                if(fabs(lanmeta1_1)>=0.1)                                    //熵修正
                {
                        lanmeta1_1=fabs(lanmeta1_1); }
                else
                  {
                        lanmeta1_1=(lanmeta1_1*lanmeta1_1+0.01)/0.2; }
                if(fabs(lanmeta2_1)>=0.1)
                {
                        lanmeta2_1=fabs(lanmeta2_1); }
                else
                  {
                        lanmeta2_1=(lanmeta2_1*lanmeta2_1+0.01)/0.2;}
                if(fabs(lanmeta3_1)>=0.1)
                {
                        lanmeta3_1=fabs(lanmeta3_1); }
                else
                  {
                        lanmeta3_1=(lanmeta3_1*lanmeta3_1+0.01)/0.2;}

    beta1_1=slj_1[i][j]*lanmeta1_1*(pho1_1[i][j]-pho1_1[i-1][j]-(pre1_1[i][j]-pre1_1[i-1][j])/shengsu_1/shengsu_1);

    beta2_1=slj_1[i][j]*lanmeta3_1*(pre1_1[i][j]-pre1_1[i-1][j]+pho_1_ba*shengsu_1*(Uir_1[i][j]-Uil_1[i][j]))/(2*shengsu_1*shengs
u_1);

    beta3_1=slj_1[i][j]*lanmeta2_1*(pre1_1[i][j]-pre1_1[i-1][j]-pho_1_ba*shengsu_1*(Uir_1[i][j]-Uil_1[i][j]))/(2*shengsu_1*shengs
u_1);
                beta4_1=beta1_1+beta2_1+beta3_1;
                beta5_1=shengsu_1*(beta2_1-beta3_1);

    beta6_1=slj_1[i][j]*lanmeta1_1*pho_1_ba*(vx1_1[i][j]-vx1_1[i-1][j]-dyj_1[i][j]/slj_1[i][j]*(Uir_1[i][j]-Uil_1[i][j]));

    beta7_1=slj_1[i][j]*lanmeta1_1*pho_1_ba*(vy1_1[i][j]-vy1_1[i-1][j]-dxj_1[i][j]/slj_1[i][j]*(Uir_1[i][j]-Uil_1[i][j]));
                AQi_1[i][j][0]=beta4_1;
                AQi_1[i][j][1]=vx1_1_ba*beta4_1+dyj_1[i][j]/slj_1[i][j]*beta5_1+beta6_1;
                AQi_1[i][j][2]=vy1_1_ba*beta4_1+dxj_1[i][j]/slj_1[i][j]*beta5_1+beta7_1;

    AQi_1[i][j][3]=H_1_ba*beta4_1+U_1_ba*beta5_1+vx1_1_ba*beta6_1+vy1_1_ba*beta7_1-shengsu_1*shengsu_1*beta1_1/(gam
a-1);
             }
        }
          for(j=1;j<80;j++)
        {
            for(i=1;i<261;i++)
            {
                pho_2_ba=sqrt(pho2_1[i-1][j]*pho2_1[i][j]);                    //Roe 平均定义

    vx2_2_ba=(vx2_1[i-1][j]*sqrt(pho2_1[i-1][j])+vx2_1[i][j]*sqrt(pho2_1[i][j]))/(sqrt(pho2_1[i-1][j])+sqrt(pho2_1[i][j]));
```

```
vy2_2_ba=(vy2_1[i-1][j]*sqrt(pho2_1[i-1][j])+vy2_1[i][j]*sqrt(pho2_1[i][j]))/(sqrt(pho2_1[i-1][j])+sqrt(pho2_1[i][j]));

H_2_ba=(H_2[i-1][j]*sqrt(pho2_1[i-1][j])+H_2[i][j]*sqrt(pho2_1[i][j]))/(sqrt(pho2_1[i-1][j])+sqrt(pho2_1[i][j]));

U_2_ba=(Uil_2[i][j]*sqrt(pho2_1[i-1][j])+Uir_2[i][j]*sqrt(pho2_1[i][j]))/(sqrt(pho2_1[i-1][j])+sqrt(pho2_1[i][j]));
                        shengsu_2=sqrt((gama-1)*(H_2_ba-(vx2_2_ba*vx2_2_ba+vy2_2_ba*vy2_2_ba)/2));
                        lanmeta1_2=U_2_ba;
                        lanmeta2_2=U_2_ba-shengsu_2;
                        lanmeta3_2=U_2_ba+shengsu_2;
                        if(fabs(lanmeta1_2)>=0.1)                                       //熵修正
                        {
                                lanmeta1_2=fabs(lanmeta1_2); }
                        else
                          {
                                lanmeta1_2=(lanmeta1_2*lanmeta1_2+0.01)/0.2; }
                        if(fabs(lanmeta2_2)>=0.1)
                        {
                                lanmeta2_2=fabs(lanmeta2_2); }
                        else
                          {
                                lanmeta2_2=(lanmeta2_2*lanmeta2_2+0.01)/0.2;}
                        if(fabs(lanmeta3_2)>=0.1)
                        {
                                lanmeta3_2=fabs(lanmeta3_2); }
                        else
                          {
                                lanmeta3_2=(lanmeta3_2*lanmeta3_2+0.01)/0.2;}

beta1_2=slj_2[i][j]*lanmeta1_2*(pho2_1[i][j]-pho2_1[i-1][j]-(pre2_1[i][j]-pre2_1[i-1][j])/shengsu_2/shengsu_2);

beta2_2=slj_2[i][j]*lanmeta3_2*(pre2_1[i][j]-pre2_1[i-1][j]+pho_2_ba*shengsu_2*(Uir_2[i][j]-Uil_2[i][j]))/(2*shengsu_2*shengs
u_2);

beta3_2=slj_2[i][j]*lanmeta2_2*(pre2_1[i][j]-pre2_1[i-1][j]-pho_2_ba*shengsu_2*(Uir_2[i][j]-Uil_2[i][j]))/(2*shengsu_2*shengs
u_2);
                        beta4_2=beta1_2+beta2_2+beta3_2;
                        beta5_2=shengsu_2*(beta2_2-beta3_2);

beta6_2=slj_2[i][j]*lanmeta1_2*pho_2_ba*(vx2_1[i][j]-vx2_1[i-1][j]-dyj_2[i][j]/slj_2[i][j]*(Uir_2[i][j]-Uil_2[i][j]));

beta7_2=slj_2[i][j]*lanmeta1_2*pho_2_ba*(vy2_1[i][j]-vy2_1[i-1][j]-dxj_2[i][j]/slj_2[i][j]*(Uir_2[i][j]-Uil_2[i][j]));
                        AQi_2[i][j][0]=beta4_2;
                        AQi_2[i][j][1]=vx2_2_ba*beta4_2+dyj_2[i][j]/slj_2[i][j]*beta5_2+beta6_2;
                        AQi_2[i][j][2]=vy2_2_ba*beta4_2+dxj_2[i][j]/slj_2[i][j]*beta5_2+beta7_2;

AQi_2[i][j][3]=H_2_ba*beta4_2+U_2_ba*beta5_2+vx2_2_ba*beta6_2+vy2_2_ba*beta7_2-shengsu_2*shengsu_2*beta1_2/(gam
a-1);
                }
          }
          for(j=1;j<71;j++)
          {
                for(i=1;i<330;i++)
                {
                        pho_1_ba=sqrt(pho1_1[i][j-1]*pho1_1[i][j]);

vx1_1_ba=(vx1_1[i][j-1]*sqrt(pho1_1[i][j-1])+vx1_1[i][j]*sqrt(pho1_1[i][j]))/(sqrt(pho1_1[i][j-1])+sqrt(pho1_1[i][j]));

vy1_1_ba=(vy1_1[i][j-1]*sqrt(pho1_1[i][j-1])+vy1_1[i][j]*sqrt(pho1_1[i][j]))/(sqrt(pho1_1[i][j-1])+sqrt(pho1_1[i][j]));

H_1_ba=(H_1[i][j-1]*sqrt(pho1_1[i][j-1])+H_1[i][j]*sqrt(pho1_1[i][j]))/(sqrt(pho1_1[i][j-1])+sqrt(pho1_1[i][j]));

U_1_ba=(Ujl_1[i][j]*sqrt(pho1_1[i][j-1])+Ujr_1[i][j]*sqrt(pho1_1[i][j]))/(sqrt(pho1_1[i][j-1])+sqrt(pho1_1[i][j]));
                        shengsu_1=sqrt((gama-1)*(H_1_ba-(vx1_1_ba*vx1_1_ba+vy1_1_ba*vy1_1_ba)/2));
                        lanmeta1_1=U_1_ba;
                        lanmeta2_1=U_1_ba-shengsu_1;
                        lanmeta3_1=U_1_ba+shengsu_1;
                        if(fabs(lanmeta1_1)>=0.1)
                        {
                                lanmeta1_1=fabs(lanmeta1_1); }
                        else
                          {
                                lanmeta1_1=(lanmeta1_1*lanmeta1_1+0.01)/0.2; }
                        if(fabs(lanmeta2_1)>=0.1)
                        {
                                lanmeta2_1=fabs(lanmeta2_1); }
                        else
                          {
                                lanmeta2_1=(lanmeta2_1*lanmeta2_1+0.01)/0.2;}
                        if(fabs(lanmeta3_1)>=0.1)
```

```
                {
                        lanmeta3_1=fabs(lanmeta3_1); }
                else
                  {
                        lanmeta3_1=(lanmeta3_1*lanmeta3_1+0.01)/0.2;}

        beta1_1=sli_1[i][j]*lanmeta1_1*(pho1_1[i][j]-pho1_1[i][j-1]-(pre1_1[i][j]-pre1_1[i][j-1])/shengsu_1/shengsu_1);

        beta2_1=sli_1[i][j]*lanmeta3_1*(pre1_1[i][j]-pre1_1[i][j-1]+pho_1_ba*shengsu_1*(Ujr_1[i][j]-Ujl_1[i][j]))/(2*shengsu_1*shengs
u_1);

        beta3_1=sli_1[i][j]*lanmeta2_1*(pre1_1[i][j]-pre1_1[i][j-1]-pho_1_ba*shengsu_1*(Ujr_1[i][j]-Ujl_1[i][j]))/(2*shengsu_1*shengs
u_1);
                beta4_1=beta1_1+beta2_1+beta3_1;
                beta5_1=shengsu_1*(beta2_1-beta3_1);

        beta6_1=sli_1[i][j]*lanmeta1_1*pho_1_ba*(vx1_1[i][j]-vx1_1[i][j-1]-dyi_1[i][j]/sli_1[i][j]*(Ujr_1[i][j]-Ujl_1[i][j]));

        beta7_1=sli_1[i][j]*lanmeta1_1*pho_1_ba*(vy1_1[i][j]-vy1_1[i][j-1]-dxi_1[i][j]/sli_1[i][j]*(Ujr_1[i][j]-Ujl_1[i][j]));
                AQj_1[i][j][0]=beta4_1;
                AQj_1[i][j][1]=vx1_1_ba*beta4_1+dyi_1[i][j]/sli_1[i][j]*beta5_1+beta6_1;
                AQj_1[i][j][2]=vy1_1_ba*beta4_1+dxi_1[i][j]/sli_1[i][j]*beta5_1+beta7_1;

        AQj_1[i][j][3]=H_1_ba*beta4_1+U_1_ba*beta5_1+vx1_1_ba*beta6_1+vy1_1_ba*beta7_1-shengsu_1*shengsu_1*beta1_1/(gam
a-1);
                }
          }
            for(j=1;j<81;j++)
          {
                for(i=1;i<260;i++)
                {
                        pho_2_ba=sqrt(pho2_1[i][j-1]*pho2_1[i][j]);

        vx2_2_ba=(vx2_1[i][j-1]*sqrt(pho2_1[i][j-1])+vx2_1[i][j]*sqrt(pho2_1[i][j]))/(sqrt(pho2_1[i][j-1])+sqrt(pho2_1[i][j]));

        vy2_2_ba=(vy2_1[i][j-1]*sqrt(pho2_1[i][j-1])+vy2_1[i][j]*sqrt(pho2_1[i][j]))/(sqrt(pho2_1[i][j-1])+sqrt(pho2_1[i][j]));

        H_2_ba=(H_2[i][j-1]*sqrt(pho2_1[i][j-1])+H_2[i][j]*sqrt(pho2_1[i][j]))/(sqrt(pho2_1[i][j-1])+sqrt(pho2_1[i][j]));

        U_2_ba=(Ujl_2[i][j]*sqrt(pho2_1[i][j-1])+Ujr_2[i][j]*sqrt(pho2_1[i][j]))/(sqrt(pho2_1[i][j-1])+sqrt(pho2_1[i][j]));
                        shengsu_2=sqrt((gama-1)*(H_2_ba-(vx2_2_ba*vx2_2_ba+vy2_2_ba*vy2_2_ba)/2));
                        lanmeta1_2=U_2_ba;
                        lanmeta2_2=U_2_ba-shengsu_2;
                        lanmeta3_2=U_2_ba+shengsu_2;
                        if(fabs(lanmeta1_2)>=0.1)
                        {
                                lanmeta1_2=fabs(lanmeta1_2); }
                        else
                          {
                                lanmeta1_2=(lanmeta1_2*lanmeta1_2+0.01)/0.2; }
                        if(fabs(lanmeta2_2)>=0.1)
                        {
                                lanmeta2_2=fabs(lanmeta2_2); }
                        else
                          {
                                lanmeta2_2=(lanmeta2_2*lanmeta2_2+0.01)/0.2;}
                        if(fabs(lanmeta3_2)>=0.1)
                        {
                                lanmeta3_2=fabs(lanmeta3_2); }
                        else
                          {
                                lanmeta3_2=(lanmeta3_2*lanmeta3_2+0.01)/0.2;}

        beta1_2=sli_2[i][j]*lanmeta1_2*(pho2_1[i][j]-pho2_1[i][j-1]-(pre2_1[i][j]-pre2_1[i][j-1])/shengsu_2/shengsu_2);

        beta2_2=sli_2[i][j]*lanmeta3_2*(pre2_1[i][j]-pre2_1[i][j-1]+pho_2_ba*shengsu_2*(Ujr_2[i][j]-Ujl_2[i][j]))/(2*shengsu_2*shengs
u_2);

        beta3_2=sli_2[i][j]*lanmeta2_2*(pre2_1[i][j]-pre2_1[i][j-1]-pho_2_ba*shengsu_2*(Ujr_2[i][j]-Ujl_2[i][j]))/(2*shengsu_2*shengs
u_2);
                        beta4_2=beta1_2+beta2_2+beta3_2;
                        beta5_2=shengsu_2*(beta2_2-beta3_2);

        beta6_2=sli_2[i][j]*lanmeta1_2*pho_2_ba*(vx2_1[i][j]-vx2_1[i][j-1]-dyi_2[i][j]/sli_2[i][j]*(Ujr_2[i][j]-Ujl_2[i][j]));

        beta7_2=sli_1[i][j]*lanmeta1_2*pho_2_ba*(vy2_1[i][j]-vy2_1[i][j-1]-dxi_2[i][j]/sli_2[i][j]*(Ujr_2[i][j]-Ujl_2[i][j]));
                        AQj_2[i][j][0]=beta4_2;
                        AQj_2[i][j][1]=vx2_2_ba*beta4_2+dyi_2[i][j]/sli_2[i][j]*beta5_2+beta6_2;
                        AQj_2[i][j][2]=vy2_2_ba*beta4_2+dxi_2[i][j]/sli_2[i][j]*beta5_2+beta7_2;
```

```
        AQj_2[i][j][3]=H_2_ba*beta4_2+U_2_ba*beta5_2+vx2_2_ba*beta6_2+vy2_2_ba*beta7_2-shengsu_2*shengsu_2*beta1_2/(gam
a-1);
                        }
                    }
                    maxflux=0;
                maxi=0;
                maxj=0;
                max=0;
                maxflux2=0;
                maxi2=0;
                maxj2=0;
                max2=0;
                    for(j=1;j<70;j++)
                    {
                        for(i=1;i<330;i++)
                        {

        Flux_1[i][j][0]=-(Fil_1[i][j][0]+Fir_1[i][j][0]-AQi_1[i][j][0])/2+(Fil_1[i+1][j][0]+Fir_1[i+1][j][0]-AQi_1[i+1][j][0])/2-(Fjl_1[i][j][
0]+Fjr_1[i][j][0]-AQj_1[i][j][0])/2+(Fjl_1[i][j+1][0]+Fjr_1[i][j+1][0]-AQj_1[i][j+1][0])/2;

        Flux_1[i][j][1]=-(Fil_1[i][j][1]+Fir_1[i][j][1]-AQi_1[i][j][1])/2+(Fil_1[i+1][j][1]+Fir_1[i+1][j][1]-AQi_1[i+1][j][1])/2-(Fjl_1[i][j][
1]+Fjr_1[i][j][1]-AQj_1[i][j][1])/2+(Fjl_1[i][j+1][1]+Fjr_1[i][j+1][1]-AQj_1[i][j+1][1])/2;

        Flux_1[i][j][2]=-(Fil_1[i][j][2]+Fir_1[i][j][2]-AQi_1[i][j][2])/2+(Fil_1[i+1][j][2]+Fir_1[i+1][j][2]-AQi_1[i+1][j][2])/2-(Fjl_1[i][j][
2]+Fjr_1[i][j][2]-AQj_1[i][j][2])/2+(Fjl_1[i][j+1][2]+Fjr_1[i][j+1][2]-AQj_1[i][j+1][2])/2;

        Flux_1[i][j][3]=-(Fil_1[i][j][3]+Fir_1[i][j][3]-AQi_1[i][j][3])/2+(Fil_1[i+1][j][3]+Fir_1[i+1][j][3]-AQi_1[i+1][j][3])/2-(Fjl_1[i][j][
3]+Fjr_1[i][j][3]-AQj_1[i][j][3])/2+(Fjl_1[i][j+1][3]+Fjr_1[i][j+1][3]-AQj_1[i][j+1][3])/2;
                            if(Flux_1[i][j][0]>maxflux)
            {maxflux=Flux_1[i][j][0];
               maxi=i;
               maxj=j;
               max=1;
            }
            if(Flux_1[i][j][1]>maxflux)
            {maxflux=Flux_1[i][j][1];
            maxi=i;
               maxj=j;
               max=2;
            }
            if(Flux_1[i][j][2]>maxflux)
            {maxflux=Flux_1[i][j][2];
            maxi=i;
               maxj=j;
               max=3;
            }
            if(Flux_1[i][j][3]>maxflux)
            {maxflux=Flux_1[i][j][3];
            maxi=i;
               maxj=j;
               max=4;
            }
                        }
                    }
                    for(j=1;j<80;j++)
                    {
                        for(i=1;i<260;i++)
                        {

        Flux_2[i][j][0]=-(Fil_2[i][j][0]+Fir_2[i][j][0]-AQi_2[i][j][0])/2+(Fil_2[i+1][j][0]+Fir_2[i+1][j][0]-AQi_2[i+1][j][0])/2-(Fjl_2[i][j][
0]+Fjr_2[i][j][0]-AQj_2[i][j][0])/2+(Fjl_2[i][j+1][0]+Fjr_2[i][j+1][0]-AQj_2[i][j+1][0])/2;

        Flux_2[i][j][1]=-(Fil_2[i][j][1]+Fir_2[i][j][1]-AQi_2[i][j][1])/2+(Fil_2[i+1][j][1]+Fir_2[i+1][j][1]-AQi_2[i+1][j][1])/2-(Fjl_2[i][j][
1]+Fjr_2[i][j][1]-AQj_2[i][j][1])/2+(Fjl_2[i][j+1][1]+Fjr_2[i][j+1][1]-AQj_2[i][j+1][1])/2;

        Flux_2[i][j][2]=-(Fil_2[i][j][2]+Fir_2[i][j][2]-AQi_2[i][j][2])/2+(Fil_2[i+1][j][2]+Fir_2[i+1][j][2]-AQi_2[i+1][j][2])/2-(Fjl_2[i][j][
2]+Fjr_2[i][j][2]-AQj_2[i][j][2])/2+(Fjl_2[i][j+1][2]+Fjr_2[i][j+1][2]-AQj_2[i][j+1][2])/2;

        Flux_2[i][j][3]=-(Fil_2[i][j][3]+Fir_2[i][j][3]-AQi_2[i][j][3])/2+(Fil_2[i+1][j][3]+Fir_2[i+1][j][3]-AQi_2[i+1][j][3])/2-(Fjl_2[i][j][
3]+Fjr_2[i][j][3]-AQj_2[i][j][3])/2+(Fjl_2[i][j+1][3]+Fjr_2[i][j+1][3]-AQj_2[i][j+1][3])/2;
                            if(Flux_2[i][j][0]>maxflux2)
            {maxflux2=Flux_2[i][j][0];
               maxi2=i;
               maxj2=j;
               max2=1;
            }
            if(Flux_2[i][j][1]>maxflux2)
            {maxflux2=Flux_2[i][j][1];
            maxi2=i;
```

```
        maxj2=j;
        max2=2;
    }
    if(Flux_2[i][j][2]>maxflux2)
    {maxflux2=Flux_2[i][j][2];
    maxi2=i;
        maxj2=j;
        max2=3;
    }
    if(Flux_2[i][j][3]>maxflux2)
    {maxflux2=Flux_2[i][j][3];
    maxi2=i;
        maxj2=j;
        max2=4;
    }
                }
            }
            }

void iteration()
{
  for(j=1;j<70;j++)
  {
        for(i=1;i<330;i++)
        {
            Q_1[i][j][0]=pho1_1[i][j];
            Q_1[i][j][1]=pho1_1[i][j]*vx1_1[i][j];
            Q_1[i][j][2]=pho1_1[i][j]*vy1_1[i][j];
            Q_1[i][j][3]=pre1_1[i][j]/(gama-1)+0.5*pho1_1[i][j]*(vx1_1[i][j]*vx1_1[i][j]+vy1_1[i][j]*vy1_1[i][j]);
            }
        }

            for(j=1;j<80;j++)
  {
        for(i=1;i<260;i++)
        {
            Q_2[i][j][0]=pho2_1[i][j];
            Q_2[i][j][1]=pho2_1[i][j]*vx2_1[i][j];
            Q_2[i][j][2]=pho2_1[i][j]*vy2_1[i][j];
            Q_2[i][j][3]=pre2_1[i][j]/(gama-1)+0.5*pho2_1[i][j]*(vx2_1[i][j]*vx2_1[i][j]+vy2_1[i][j]*vy2_1[i][j]);
            }
        }
        for(j=1;j<70;j++)
        {
            for(i=1;i<330;i++)
            {
                tyj=0.5*(dyj_1[i][j]+dyj_1[i+1][j]);
                txj=0.5*(dxj_1[i][j]+dxj_1[i+1][j]);
                tslj=0.5*(slj_1[i][j]+slj_1[i+1][j]);
                tyi=0.5*(dyi_1[i][j]+dyi_1[i][j+1]);
                txi=0.5*(dxi_1[i][j]+dxi_1[i][j+1]);
                tsli=0.5*(sli_1[i][j]+sli_1[i][j+1]);
                vj=tyj*vx1_1[i][j]+txj*vy1_1[i][j];
                vi=tyi*vx1_1[i][j]+txi*vy1_1[i][j];
                sonic=sqrt(gama*pre1_1[i][j]/pho1_1[i][j]);
                chvel=fabs(vj)+fabs(vi)+sonic*(tslj+tsli);
                dt=cfl/chvel;

                Q_1[i][j][0]=Q_1[i][j][0]-dt*Flux_1[i][j][0];                   //迭代求解下一步 Q
                Q_1[i][j][1]=Q_1[i][j][1]-dt*Flux_1[i][j][1];
                Q_1[i][j][2]=Q_1[i][j][2]-dt*Flux_1[i][j][2];
                Q_1[i][j][3]=Q_1[i][j][3]-dt*Flux_1[i][j][3];
                pho1_1[i][j]=Q_1[i][j][0];
                vx1_1[i][j]=Q_1[i][j][1]/Q_1[i][j][0];
                vy1_1[i][j]=Q_1[i][j][2]/Q_1[i][j][0];
                pre1_1[i][j]=(gama-1)*(Q_1[i][j][3]-0.5*pho1_1[i][j]*(vx1_1[i][j]*vx1_1[i][j]+vy1_1[i][j]*vy1_1[i][j]));
                T1_1[i][j]=pre1_1[i][j]/pho1_1[i][j]/R;
                ma1_1[i][j]=sqrt(vx1_1[i][j]*vx1_1[i][j]+vy1_1[i][j]*vy1_1[i][j])/sqrt(gama*R*T1_1[i][j]);
            }
         }
         for(j=1;j<80;j++)
        {
            for(i=1;i<260;i++)
            {
                tyj=0.5*(dyj_2[i][j]+dyj_2[i+1][j]);
                txj=0.5*(dxj_2[i][j]+dxj_2[i+1][j]);
                tslj=0.5*(slj_2[i][j]+slj_2[i+1][j]);
                tyi=0.5*(dyi_2[i][j]+dyi_2[i][j+1]);
                txi=0.5*(dxi_2[i][j]+dxi_2[i][j+1]);
                tsli=0.5*(sli_2[i][j]+sli_2[i][j+1]);
```

```
                vj=tyj*vx2_1[i][j]+txj*vy2_1[i][j];
                vi=tyi*vx2_1[i][j]+txi*vy2_1[i][j];
                sonic=sqrt(gama*pre2_1[i][j]/pho2_1[i][j]);
                chvel=fabs(vj)+fabs(vi)+sonic*(tslj+tsli);
                dt=cfl/chvel;
                Q_2[i][j][0]=Q_2[i][j][0]-dt*Flux_2[i][j][0];                    //迭代求解下一步 Q
                Q_2[i][j][1]=Q_2[i][j][1]-dt*Flux_2[i][j][1];
                Q_2[i][j][2]=Q_2[i][j][2]-dt*Flux_2[i][j][2];
                Q_2[i][j][3]=Q_2[i][j][3]-dt*Flux_2[i][j][3];
                pho2_1[i][j]=Q_2[i][j][0];
                vx2_1[i][j]=Q_2[i][j][1]/Q_2[i][j][0];
                vy2_1[i][j]=Q_2[i][j][2]/Q_2[i][j][0];
                pre2_1[i][j]=(gama-1)*(Q_2[i][j][3]-0.5*pho2_1[i][j]*(vx2_1[i][j]*vx2_1[i][j]+vy2_1[i][j]*vy2_1[i][j]));
                T2_1[i][j]=pre2_1[i][j]/pho2_1[i][j]/R;
                ma2_1[i][j]=sqrt(vx2_1[i][j]*vx2_1[i][j]+vy2_1[i][j]*vy2_1[i][j])/sqrt(gama*R*T2_1[i][j]);
            }
        }
}
void record()
{
    for(j=1;j<70;j++)
    {
        for(i=1;i<330;i++)
        {
            nodesc1[i][j][0]=0.25*(nodes[i][j][0]+nodes[i+1][j][0]+nodes[i+1][j+1][0]+nodes[i][j+1][0]);
            nodesc1[i][j][1]=0.25*(nodes[i][j][1]+nodes[i+1][j][1]+nodes[i+1][j+1][1]+nodes[i][j+1][1]);
        }
    }
    for(j=1;j<70;j++)
    {
        nodesc1[0][j][0]=0.5*(nodes[1][j][0]+nodes[1][j+1][0]);
        nodesc1[0][j][1]=0.5*(nodes[1][j][1]+nodes[1][j+1][1]);
        nodesc1[330][j][0]=0.5*(nodes[330][j][0]+nodes[330][j+1][0]);
        nodesc1[330][j][1]=0.5*(nodes[330][j][1]+nodes[330][j+1][1]);
    }
    for(i=1;i<330;i++)
    {
        nodesc1[i][0][0]=0.5*(nodes[i][0][0]+nodes[i+1][0][0]);
        nodesc1[i][0][1]=0.5*(nodes[i][0][1]+nodes[i+1][0][1]);
        nodesc1[i][70][0]=0.5*(nodes[i][70][0]+nodes[i+1][70][0]);
        nodesc1[i][70][1]=0.5*(nodes[i][70][1]+nodes[i+1][70][1]);
    }
    nodesc1[0][0][0]=nodes[1][1][0];
    nodesc1[0][0][1]=nodes[1][1][1];
    nodesc1[330][0][0]=nodes[330][1][0];
    nodesc1[330][0][1]=nodes[330][1][1];
    nodesc1[330][70][0]=nodes[330][70][0];
    nodesc1[330][70][1]=nodes[330][70][1];
    nodesc1[0][70][0]=nodes[1][70][0];
    nodesc1[0][70][1]=nodes[1][70][1];
    for(j=1;j<70;j++)
    {
        pho1_1[0][j]=0.5*(pho1_1[0][j]+pho1_1[1][j]);
        vx1_1[0][j]=0.5*(vx1_1[0][j]+vx1_1[1][j]);
        vy1_1[0][j]=0.5*(vy1_1[0][j]+vy1_1[1][j]);
        pre1_1[0][j]=0.5*(pre1_1[0][j]+pre1_1[1][j]);
        pho1_1[330][j]=0.5*(pho1_1[330][j]+pho1_1[259][j]);
        vx1_1[330][j]=0.5*(vx1_1[330][j]+vx1_1[259][j]);
        vy1_1[330][j]=0.5*(vy1_1[330][j]+vy1_1[259][j]);
        pre1_1[330][j]=0.5*(pre1_1[330][j]+pre1_1[259][j]);
    }
    for(i=1;i<330;i++)
    {
        pho1_1[i][0]=0.5*(pho1_1[i][0]+pho1_1[i][1]);
        vx1_1[i][0]=0.5*(vx1_1[i][0]+vx1_1[i][1]);
        vy1_1[i][0]=0.5*(vy1_1[i][0]+vy1_1[i][1]);
        pre1_1[i][0]=0.5*(pre1_1[i][0]+pre1_1[i][1]);
        pho1_1[i][70]=0.5*(pho1_1[i][70]+pho1_1[i][69]);
        vx1_1[i][70]=0.5*(vx1_1[i][70]+vx1_1[i][69]);
        vy1_1[i][70]=0.5*(vy1_1[i][70]+vy1_1[i][69]);
        pre1_1[i][70]=0.5*(pre1_1[i][70]+pre1_1[i][69]);
    }
    pho1_1[0][0]=pho1_1[1][0];
    vx1_1[0][0]=vx1_1[1][0];
    vy1_1[0][0]=vy1_1[1][0];
    pre1_1[0][0]=pre1_1[1][0];

    pho1_1[0][70]=pho1_1[1][70];
    vx1_1[0][70]=vx1_1[1][70];
    vy1_1[0][70]=vy1_1[1][70];
```

```
pre1_1[0][70]=pre1_1[1][70];

pho1_1[330][0]=pho1_1[259][0];
vx1_1[330][0]=vx1_1[259][0];
vy1_1[330][0]=vy1_1[259][0];
pre1_1[330][0]=pre1_1[259][0];

pho1_1[330][70]=pho1_1[259][70];
vx1_1[330][70]=vx1_1[259][70];
vy1_1[330][70]=vy1_1[259][70];
pre1_1[330][70]=pre1_1[259][70];
for(j=1;j<80;j++)
{
	for(i=1;i<260;i++)
	{
		nodesc2[i][j][0]=0.25*(nodes2[i][j][0]+nodes2[i+1][j][0]+nodes2[i+1][j+1][0]+nodes2[i][j+1][0]);
		nodesc2[i][j][1]=0.25*(nodes2[i][j][1]+nodes2[i+1][j][1]+nodes2[i+1][j+1][1]+nodes2[i][j+1][1]);
	}
}
for(j=1;j<80;j++)
{
	nodesc2[0][j][0]=0.5*(nodes2[1][j][0]+nodes2[1][j+1][0]);
	nodesc2[0][j][1]=0.5*(nodes2[1][j][1]+nodes2[1][j+1][1]);
	nodesc2[260][j][0]=0.5*(nodes2[260][j][0]+nodes2[260][j+1][0]);
	nodesc2[260][j][1]=0.5*(nodes2[260][j][1]+nodes2[260][j+1][1]);
}
for(i=1;i<260;i++)
{
	nodesc2[i][0][0]=0.5*(nodes2[i][0][0]+nodes2[i+1][0][0]);
	nodesc2[i][0][1]=0.5*(nodes2[i][0][1]+nodes2[i+1][0][1]);
	nodesc2[i][80][0]=0.5*(nodes2[i][80][0]+nodes2[i+1][80][0]);
	nodesc2[i][80][1]=0.5*(nodes2[i][80][1]+nodes2[i+1][80][1]);
}
nodesc2[0][0][0]=nodes2[1][1][0];
nodesc2[0][0][1]=nodes2[1][1][1];
nodesc2[260][0][0]=nodes2[260][1][0];
nodesc2[260][0][1]=nodes2[260][1][1];
nodesc2[260][80][0]=nodes2[260][80][0];
nodesc2[260][80][1]=nodes2[260][80][1];
nodesc2[0][80][0]=nodes2[1][80][0];
nodesc2[0][80][1]=nodes2[1][80][1];
for(j=1;j<80;j++)
{
	pho2_1[0][j]=0.5*(pho2_1[0][j]+pho2_1[1][j]);
	vx2_1[0][j]=0.5*(vx2_1[0][j]+vx2_1[1][j]);
	vy2_1[0][j]=0.5*(vy2_1[0][j]+vy2_1[1][j]);
	pre2_1[0][j]=0.5*(pre2_1[0][j]+pre2_1[1][j]);
	pho2_1[260][j]=0.5*(pho2_1[260][j]+pho2_1[259][j]);
	vx2_1[260][j]=0.5*(vx2_1[260][j]+vx2_1[259][j]);
	vy2_1[260][j]=0.5*(vy2_1[260][j]+vy2_1[259][j]);
	pre2_1[260][j]=0.5*(pre2_1[260][j]+pre2_1[259][j]);
}
for(i=1;i<260;i++)
{
	pho2_1[i][0]=0.5*(pho2_1[i][0]+pho2_1[i][1]);
	vx2_1[i][0]=0.5*(vx2_1[i][0]+vx2_1[i][1]);
	vy2_1[i][0]=0.5*(vy2_1[i][0]+vy2_1[i][1]);
	pre2_1[i][0]=0.5*(pre2_1[i][0]+pre2_1[i][1]);
	pho2_1[i][80]=0.5*(pho2_1[i][80]+pho2_1[i][79]);
	vx2_1[i][80]=0.5*(vx2_1[i][80]+vx2_1[i][79]);
	vy2_1[i][80]=0.5*(vy2_1[i][80]+vy2_1[i][79]);
	pre2_1[i][80]=0.5*(pre2_1[i][80]+pre2_1[i][79]);
}
pho2_1[0][0]=pho2_1[1][0];
vx2_1[0][0]=vx2_1[1][0];
vy2_1[0][0]=vy2_1[1][0];
pre2_1[0][0]=pre2_1[1][0];

pho2_1[0][80]=pho2_1[1][80];
vx2_1[0][80]=vx2_1[1][80];
vy2_1[0][80]=vy2_1[1][80];
pre2_1[0][80]=pre2_1[1][80];

pho2_1[260][0]=pho2_1[259][0];
vx2_1[260][0]=vx2_1[259][0];
vy2_1[260][0]=vy2_1[259][0];
pre2_1[260][0]=pre2_1[259][0];

pho2_1[260][80]=pho2_1[259][80];
vx2_1[260][80]=vx2_1[259][80];
```

```
                vy2_1[260][80]=vy2_1[259][80];
                pre2_1[260][80]=pre2_1[259][80];
        }
        void expo()
    {
          fq=fopen("wangge1.plt","w");
    fprintf(fq,"Title=\"NOZZLE\"\nVariables=\"x\",\"y\"\nZone T=\"NOZZLE\" i=331,j=71,f=point \n");
                for(j=0;j<71;j++)
        {
                for(i=0;i<331;i++)
                {
                        fprintf(fq,"%.5f         %.5f\n",nodes[i][j][0],nodes[i][j][1]);
                }
        }
         fq=fopen("wangge2.plt","w");
          fprintf(fq,"Title=\"NOZZLE\"\nVariables=\"x\",\"y\"\nZone T=\"NOZZLE\" i=261,j=81,f=point \n");
for(i=0;i<261;i++)
        {
                for(j=0;j<81;j++)
                {
                        fprintf(fq,"%.10f       %.10f\n",nodes2[i][j][0],nodes2[i][j][1]);
                }
        }
    fp=fopen("1.txt","w");
                fprintf(fp,"x                       ,y                      pho1_1[i][j],       vx1_1[i][j],        vy1_1[i][j],        pre1_1[i][j],
T1_1[i][j],         ma1_1[i][j]\n");
                for(i=1;i<330;i++)
        {
                for(j=1;j<70;j++)
                {
                        T1_1[i][j]=pre1_1[i][j]/R/pho1_1[i][j];
                        ma1_1[i][j]=sqrt(vx1_1[i][j]*vx1_1[i][j]+vy1_1[i][j]*vy1_1[i][j])/sqrt(gama*R*T1_1[i][j]);

        fprintf(fp,"%.10f       %.10f       %.10f       %.10f       %.10f       %.10f       %.10f       %.10f\n",nodes[i][j][0],nodes[i][j][1],pho1
_1[i][j],vx1_1[i][j],vy1_1[i][j],pre1_1[i][j],T1_1[i][j],ma1_1[i][j]);
                }
        }
                fq=fopen("2.txt","w");
        fprintf(fq,"x                       ,y                      pho1_1[i][j],       vx1_1[i][j],        vy1_1[i][j],        pre1_1[i][j],
T1_1[i][j],         ma1_1[i][j]\n");
                for(i=1;i<260;i++)
        {
                for(j=1;j<80;j++)
                {

        fprintf(fq,"%.10f       %.10f       %.10f       %.10f       %.10f       %.10f       %.10f       %.10f\n",nodes2[i][j][0],nodes2[i][j][1],ph
o2_1[i][j],vx2_1[i][j],vy2_1[i][j],pre2_1[i][j],T2_1[i][j],ma2_1[i][j]);
                }
        }
                fr=fopen("WYplotflow.plt","w");
                fprintf(fr,"Title=\"NOZZLE\"\nVariables=\"x\",\"y\",\"dens\",\"velx\",\"vely\",\"spre\",\"ttem\",\"mach\"\nZone
T=\"NOZZLE\" i=331,j=71,f=point \n");
        for(j=0;j<71;j++)
        {
                for(i=0;i<331;i++)
                {
                        T1_1[i][j]=pre1_1[i][j]/R/pho1_1[i][j];
                        ma1_1[i][j]=sqrt(vx1_1[i][j]*vx1_1[i][j]+vy1_1[i][j]*vy1_1[i][j])/sqrt(gama*R*T1_1[i][j]);

        fprintf(fr,"%.5f    %.5f    %.5f    %.5f    %.5f    %.5f    %.5f    %.5f\n",nodesc1[i][j][0],nodesc1[i][j][1],pho1_1[i][j],vx1_1[i][j],vy1
_1[i][j],pre1_1[i][j],T1_1[i][j],ma1_1[i][j]);
                }
        }
        fs=fopen("WYplotflow2.plt","w");
                fprintf(fs,"Title=\"NOZZLE\"\nVariables=\"x\",\"y\",\"dens\",\"velx\",\"vely\",\"spre\",\"ttem\",\"mach\"\nZone
T=\"NOZZLE\" i=261,j=81,f=point \n");
        for(j=0;j<81;j++)
        {
                for(i=0;i<261;i++)
                {
                        T2_1[i][j]=pre2_1[i][j]/R/pho2_1[i][j];
                        ma2_1[i][j]=sqrt(vx2_1[i][j]*vx2_1[i][j]+vy2_1[i][j]*vy2_1[i][j])/sqrt(gama*R*T2_1[i][j]);

        fprintf(fs,"%.5f    %.5f    %.5f    %.5f    %.5f    %.5f    %.5f    %.5f\n",nodesc2[i][j][0],nodesc2[i][j][1],pho2_1[i][j],vx2_1[i][j],vy2
_1[i][j],pre2_1[i][j],T2_1[i][j],ma2_1[i][j]);
                }
        }
    }
        int main()
     {
```

```
mesh_generation();
   initialize();
   area_caculate();
fg=fopen("error.txt","w");
for(k=0;k<200000;k++)
{
   printf("%d       %.15f       %.15f       %.15f       %.15f       %.15f\n",k,maxflux,maxflux2,maxflux3,maxflux4,T2_1[3][50]);
   fprintf(fg,"%d       %.15f       %.15f       %.15f       %.15f       %.15f\n",k,maxflux,maxflux2,maxflux3,maxflux4,T2_1[3][50]);
       boundary_conditions();
   roe();
   iteration();
   }
   record();
expo();
}
```