平板稳态热传导温度分布的计算

摘要

确定一块导热平板的稳态温度分布是热力学研究中的一个重要问题. 受到现实因素的限制, 导热平板内部的温度往往无法直接测得. 当导热平板的边缘温度可测而内部温度不可测时，可以根据热力学第一定律推算出平板内部任意一点的温度. 本案例结合物理学知识，利用数值分析中线性方程组的求解方法，建立了用于求解导热平板稳态温度分布的模型，并给出了三种不同的求解方法.

本案例适用于应用理科数学类专业、新工科专业、应用物理学专业及其相关领域本科生、研究生的课程教学，如《线性代数》、《数值分析》、《现代计算方法》等，也适用于数学和交叉学科研究生开展科研训练.

关键词：稳态温度分布、线性方程组、矩阵 LU 分解、$Gauss-Seidel$ 迭代、SOR 方法

1 背景介绍

在科技飞速发展的当今年代，数学作为科技发展的核心竞争力和源动力，在与其他科学、工程技术等领域的交叉融合中起着日益重要的作用. 在这样一个数学学科深度发展和应用融合的新时代，在解决交叉科学领域中的关键问题时，应用数学的理论方法占有至关重要的作用，包括建立有效的数学模型、研制高效快速的数值算法、仿真模拟等过程.

本案例主要讨论“导热平板的稳态温度分布”这一热力学研究领域的重要问题. 在工程实践中，受现实因素的影响，往往无法直接测得导热平板内部的温度. 当导热平板的边缘温度可测而内部温度不可测时，可以根据热力学第一定律推算出平板内部任意一点的温度. 本案例结合物理学知识，利用数值分析中线性方程组的求解方法，建立了用于求解导热平板稳态温度分布的模型，并给出了三种不同的求解方法.

本案例适用于应用理科数学类专业、新工科专业、应用物理学专业及其相关领域本科生、研究生的课程教学，如《线性代数》、《数值分析》、《现代计算方法》等，也适用于数学和交叉学科研究生开展科研训练.

工程实践中，经常需要获知某导热体在稳态下的内部温度，而受限于事实情况，若是直接测量内部温度，会破坏稳态系统，如测量燃烧炉内温等. 因此，需要在仅测得边缘温度的情况下，计算求得导热体的内部诸点温度. 本案例给出了严格地推导计算过程与数值计算方法，并利用程序进行数值仿真模拟，对三种求解方法进行了对比分析，说明了模型的有效性和科学性.

2 案例内容

2.1 平板稳态热传导温度分布问题的数学描述

2.1.1 导热平板稳态热传导温度分布模型

假定可以测得平板边缘任意处的温度，那么可以对平板的边缘进行划分. 不妨设水平方向划分了 n 个分点，竖直方向划分了 m 个分点. 于是可以测得平板上、下、左、右四条边上诸点的温度，假设它们的值分别为 $T_U=(T_{u_1},T_{u_2},\cdots,T_{u_n})$；$T_D=(T_{d_1},T_{d_2},\cdots,T_{d_n})$；$T_L=(T_{l_1},T_{l_2},\cdots,T_{l_m})$；$T_R=(T_{r_1},T_{r_2},\cdots,T_{r_m})$. 于是，根据上述划分，整个平板可以看作一个 $(m+2)\times(n+2)$ 维的矩阵，矩阵上每个点的值为平板在该点的温度值. 矩阵四条边上的取值是已知的（可测量获得的），而内部诸点的取值是未知的. 记该矩阵内部的 $m\times n$ 维部分的值如下：

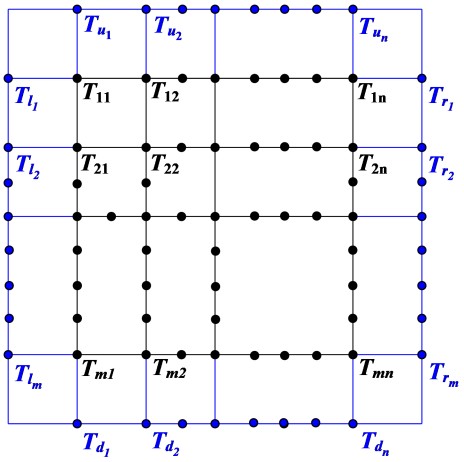

图 1: 平板区域划分

求解原问题，需要求出 $T_{11},\cdots,T_{mn}$ 共 $m\times n$ 个未知数的取值.

根据热力学第一、第二定律，在达到稳态的导热平板内，每个内点的温度应等于其临近四点温度的均值. 于是，如果记 $T_U=(T_{0,1},T_{0,2},\cdots,T_{0,n})$；$T_D=(T_{m+1,1},T_{m+1,2},\cdots,T_{m+1,n})$；$T_L=(T_{1,0},T_{2,0},\cdots,T_{m,0})$；$T_R=(T_{1,n+1},T_{2,n+1},\cdots,T_{m,n+1})$，则对于导热平板

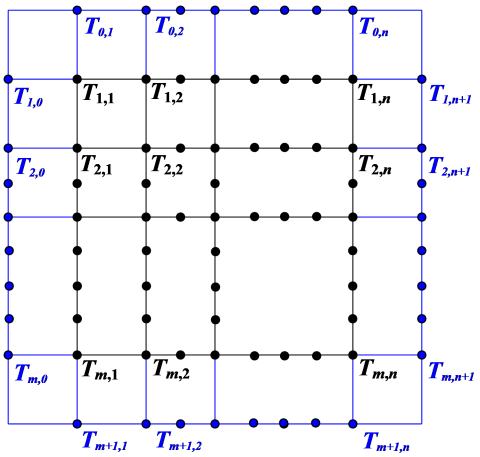

图 2: 平板划分模型

内的任意一个内点 $T_{i,j}(1\leqslant i\leqslant m,1\leqslant j\leqslant n)$，其取值都应满足

$$T_{i,j}=\frac{1}{4}\left(T_{i-1,j}+T_{i+1,j}+T_{i,j-1}+T_{i,j+1}\right) \tag{1}$$

将上式整理变形得

$$4T_{i,j}-T_{i-1,j}-T_{i+1,j}-T_{i,j-1}-T_{i,j+1}=0,1\leqslant i\leqslant m,1\leqslant j\leqslant n \tag{2}$$

于是得到关于 $T_{i,j}(1\leqslant i\leqslant m,1\leqslant j\leqslant n)$ 的 $m\times n$ 个线性方程，联立可得 $m\times n$ 维线性方程组，求解该线性方程组，即能得到诸内点 $T_{i,j}(1\leqslant i\leqslant m,1\leqslant j\leqslant n)$ 的温度值.

2.1.2 实例分析

如图 3所示，平板代表一块金属梁截面，当已知平板边缘 12 个点的温度，要求中间节点 $T_{11},T_{12},\cdots,T_{33}$ 的温度.

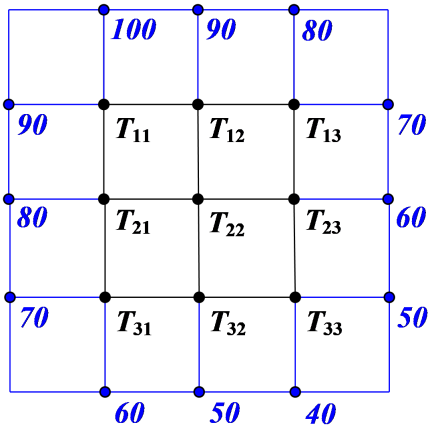

图 3: 平板温度示意图

利用公式 (1) 的建模方法，根据热力学定律，在 T_1, T_2, T_3, T_4 处，有

$$\begin{cases} T_{11} = \frac{1}{4}(90 + 100 + T_{12} + T_{21}) \\ T_{12} = \frac{1}{4}(90 + T_{11} + T_{13} + T_{22}) \\ T_{13} = \frac{1}{4}(80 + 70 + T_{12} + T_{23}) \\ T_{21} = \frac{1}{4}(80 + T_{11} + T_{22} + T_{31}) \\ T_{22} = \frac{1}{4}(T_{12} + T_{21} + T_{23} + T_{32}) \\ T_{23} = \frac{1}{4}(60 + T_{13} + T_{22} + T_{33}) \\ T_{31} = \frac{1}{4}(70 + 60 + T_{21} + T_{32}) \\ T_{32} = \frac{1}{4}(50 + T_{22} + T_{31} + T_{33}) \\ T_{33} = \frac{1}{4}(50 + 40 + T_{23} + T_{32}) \end{cases}$$

如公式 (2)，经过整理变形，得到如下线性方程组

$$\begin{cases} 4T_{11} - T_{12} - T_{21} = 190 \\ -T_{11} + 4T_{12} - T_{13} - T_{22} = 90 \\ -T_{12} + 4T_{13} - T_{23} = 150 \\ -T_{11} + 4T_{21} - T_{22} - T_{31} = 80 \\ -T_{12} - T_{21} + 4T_{22} - T_{23} - T_{32} = 0 \\ -T_{13} - T_{22} + 4T_{23} - T_{33} = 60 \\ -T_{21} + 4T_{31} - T_{32} = 130 \\ -T_{22} - T_{31} + 4T_{32} - T_{33} = 50 \\ -T_{23} - T_{32} + T_{33} = 90 \end{cases}$$

写成矩阵形式为

$$\begin{pmatrix} 4 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 4 & -1 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 4 & 0 & 0 & -1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 4 & -1 & 0 & -1 & 0 & 0 \\ 0 & -1 & 0 & -1 & 4 & -1 & 0 & -1 & 0 \\ 0 & 0 & -1 & 0 & -1 & 4 & 0 & 0 & -1 \\ 0 & 0 & 0 & -1 & 0 & 0 & 4 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & -1 & 4 & -1 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 4 \end{pmatrix} \begin{pmatrix} T_{11} \\ T_{12} \\ T_{13} \\ T_{21} \\ T_{22} \\ T_{23} \\ T_{31} \\ T_{32} \\ T_{33} \end{pmatrix} = \begin{pmatrix} 190 \\ 90 \\ 150 \\ 80 \\ 0 \\ 60 \\ 130 \\ 50 \\ 90 \end{pmatrix}$$

于是原问题转化为求解上述线性方程组.

2.2 数值算法、算例及结论

2.2.1 数值算法

分别使用矩阵 LU 分解方法、$Jacobi$ 迭代法、$Gauss-Seidel$ 迭代法和 SOR 方法对上述模型的线性方程组进行求解. 以下对上述方法的迭代过程做简要介绍.

1、直接解法——矩阵 LU 分解法：

1) 将线性方程组 $Ax=b$ 的系数矩阵 A 经过一系列行变换 $M^{(1)}M^{(2)}\cdots M^{(n-1)}$ 化为上三角型矩阵 $A^{(n)}=M^{(1)}M^{(2)}\cdots M^{(n-1)}$.

2) 因 $M^{(k)}(k=1,2,\cdots,n-1)$ 可逆且为下三角阵，于是有 $A=\left[M^{(1)}M^{(2)}\cdots M^{(n-1)}\right]^{-1}A^{(n)}$.

3) 记 $L=\left[M^{(1)}M^{(2)}\cdots M^{(n-1)}\right]^{-1}$，$U=A^{(n)}$ ，则 L 为下三角阵且对角线元素为 1，U 为上三角阵.

4) 由 $A=LU$，可以将方程组 $Ax=b$ 转化为 $Lz=b, Ux=z$ 的形式，由于 L,U 均为三角形矩阵，因此转化后的方程组的求解是显然的.

2、迭代解法——*Jacobi* 迭代法、*Gauss − Seidel* 迭代法、*SOR* 方法：

迭代法求解线性方程组的核心思想是构造使迭代式 $x^{(k)}=Tx^{(k-1)}+c$ 收敛的矩阵 T 和向量 c. 为了构造 T,c，将矩阵 A 分解为：$A=D-L-U$，其中 D 为对角阵，L 为下三角阵，U 为上三角阵. 于是有

$$(D-L-U)x=b$$

变形有

1) *Jacobi* **迭代法：** $x=D^{-1}(L+U)x+D^{-1}b$

即 $x^{(k)}=T_jx^{(k-1)}+c_j$，其中 $T_j=D^{-1}(L+U), c_j=D^{-1}b$.

写成分量形式为

$$x_i^{(k)}=\frac{1}{a_{ii}}\left(b_i-\sum_{j=1,j\neq i}^{n}a_{ij}x_j^{(k-1)}\right)\qquad i=1,2,\cdots,n$$

2) *Gauss − Seidel* **迭代法：** $x=(D-L)^{-1}Ux+(D-L)^{-1}b$

即 $x^{(k)}=T_gx^{(k-1)}+c_g$，其中 $T_g=(D-L)^{-1}U, c_g=(D-L)^{-1}b$.

写成分量形式为

$$x_i^{(k)}=\frac{1}{a_{ii}}\left(b_i-\sum_{j=1}^{i-1}a_{ij}x_j^{(k)}-\sum_{j=i+1}^{n}a_{ij}x_j^{(k-1)}\right)\qquad i=1,2,\cdots,n$$

3) *SOR* **方法：** $x=(D-\omega L)^{-1}[(1-\omega)D+\omega U]x+\omega(D-\omega L)^{-1}b,\ \omega\in(0,2)$

即 $x^{(k)}=T_\omega x^{(k-1)}+c_\omega$，其中 $T_\omega=(D-\omega L)^{-1}[(1-\omega)D+\omega U], c_\omega=\omega(D-\omega L)^{-1}b$.

写成分量形式为

$$x_i^{(k)}=(1-\omega)x_i^{(k-1)}+\frac{\omega}{a_{ii}}\left(b_i-\sum_{j=1}^{i-1}a_{ij}x_j^{(k)}-\sum_{j=i+1}^{n}a_{ij}x_j^{(k-1)}\right)\qquad i=1,2,\cdots,n$$

2.2.2 计算结果与方法对比

通过 $Matlab$ 编程实现数值算法，可以在给定导热平板边缘温度的情况下，计算出平板内部的诸点温度. 下面给出问题在程序下的求解结果（结果保留小数点后 2 位）与程序在三种方法下的运行时间.

1、$m=3, n=3$:

边缘温度：$T_U=(100,90,80)$；$T_D=(60,50,40)$；$T_L=(90,80,70)$；$T_R=(70,60,50)$

求解结果（单位：℃）:

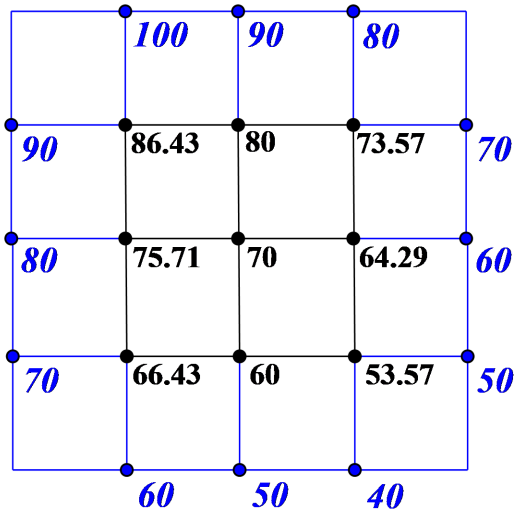

图 4: 平板温度求解结果

2、$m=50, n=50$:

边缘温度：（见附表 1：边缘温度）

求解结果（单位：℃）:（见附表 2：平板温度）

求解时间:

矩阵 LU 分解	$Jacobi$ 迭代法	$Gauss-Seidel$ 迭代法	SOR 优化方法（$\omega=1.8$）
$9.59s$	$15.56s$	$8.43s$	$0.98s$

2.3 案例优化与推广

考虑三维空间中的导热体的稳态热传导温度分布. 假定已知三维导热体（立方体）的表面温度分布，使用同样的分析方法，根据热力学定律，在稳态状态下，导热体内部点的温度应等于其附近 6 个点（上下左右前后）温度的均值. 于是可以进行相似的网格划分，将问题转换成线性方程组求解问题，进而用同样的计算方法得到三维导热体稳态状态下内部的温度分布. 该推广问题的建模求解方法留给有兴趣的读者课后完成.

3 案例小结

3.1 案例意义

本案例的教学目的是让学生多学数学、多用数学、活学数学、活用数学. 具体说来，就是让学生掌握数学思想，培养数学思维，学会数值分析建模与计算，建立数学与应用学科之桥梁；让学

生学习交叉学科知识，学会算法设计与数值模拟，以解决工程技术中的数学问题；让学生在学习与研究中，学有所获，积累成果和信心，成为优秀的、数理基础扎实的高素质科技人才.

3.2 编程心得

根据资料，$Gauss-Seidel$ 迭代法在处理大规模稀疏矩阵时具有优势. 如本模型中的系数矩阵 A，它每行至多有 5 个非零元素，在方程组阶数很大时，应满足 $Gauss-Seidel$ 迭代法的使用条件. 但事实上，最初笔者编译的 $Gauss-Seidel$ 迭代程序，在处理 50×50 问题时的运行时间长达 $131s$. 为了修正这一问题，观察 $Gauss-Seidel$ 迭代法的迭代式，发现当矩阵 A 中 0 元素较多时，迭代式中的诸多加数项恒为零，而在笔者编译的原始程序中，实质上是在每次循环中，重复计算 “$-0\times x_i$”，这些不必要的运算大大地拖慢了程序的运算时间.

在修正后的程序中，针对 $Gauss-Seidel$ 迭代法和 SOR 方法，在迭代前一步对矩阵 A 进行处理，定位矩阵 A 的全部非零项，在之后的迭代计算中，仅考虑这些非零项的加减. 这样一来，原本每次循环都需要进行的 2500×2500 次运算，简化为约 2500×5 次运算，大大减少了程序的运算量.

从这样一个经历中，可以看出，同样的数值计算方法，程序编译方式不同，呈现效果可以 “天差地别”. 因此，在学习和应用数值分析方法的过程中，如何根据实际情况编译高效的运算程序，也是应用中至关重要的一环.

4 附录：$MATLAB$ 程序

Listing 1: data2equation.m

```
function [T,A,b] = data2equation(L,V)
%根据边缘温度构造系数矩阵A和常数向量b
% L——level:水平方向边界温度，2*n矩阵
% V——vertical:竖直方向边界温度，m*2矩阵
n = size(L,2);m = size(V,1);
T = zeros(m+2,n+2); %T:金属板诸点温度示意图
A = zeros(m*n,m*n);b = zeros(m*n,1);
%A:系数矩阵；b:常数向量。（Ax=bx即为诸内点温度）
T(1,:) = [NaN,L(1,:),NaN]; T(m+2,:) = [NaN,L(2,:),NaN];
T(2:m+1,1) = V(:,1); T(2:m+1,n+2) = V(:,2);

%四个顶点的方程
A(1,1) = 4; A(1,2) = -1; A(1,n+1) = -1;b(1) = L(1,1)+V(1,1); %左上角
A(n,n) = 4; A(n,n-1) = -1; A(n,2*n) = -1; b(n) = L(1,n)+V(1,2); %右上角
A((m-1)*n+1,(m-1)*n+1) = 4; A((m-1)*n+1,(m-2)*n+1) = -1;
A((m-1)*n+1,(m-1)*n+2) = -1; b((m-1)*n+1) = L(2,1)+V(m,1); %左下角
A(m*n,m*n) = 4; A(m*n,(m-1)*n) = -1; A(m*n,m*n-1) = -1;
b(m*n) = L(2,n)+V(m,2);%右下角

%四条边上的方程
%竖直边
if m>2
    for i = 1:(m-2)
        A(i*n+1,i*n+1) = 4; A(i*n+1,(i-1)*n+1) = -1;
        A(i*n+1,(i+1)*n+1) = -1; A(i*n+1,i*n+2) = -1; b(i*n+1) = V(i+1,1);%左列
```

```
        A((i+1)*n,(i+1)*n) = 4; A((i+1)*n,i*n) = -1; A((i+1)*n,(i+2)*n) = -1;
        A((i+1)*n,(i+1)*n-1) = -1; b((i+1)*n) = V(i+1,2); %右列
    end
end
%水平边
if n>2
    for i = 2:(n-1)
        A(i,i) = 4; A(i,i-1) = -1; A(i,i+1) = -1;
        A(i,n+i) = -1; b(i) = L(1,i);%上行
        A((m-1)*n+i,(m-1)*n+i) = 4; A((m-1)*n+i,(m-1)*n+i-1) = -1;
        A((m-1)*n+i,(m-1)*n+i+1) = -1; A((m-1)*n+i,(m-2)*n+i) = -1;
        b((m-1)*n+i)=L(2,i); %下行
    end
end

%内部点
if (m>2)&(n>2)
    for i = 1:(m-2)
        for j = 2:(n-1)
            A(n*i+j,n*i+j) = 4; A(n*i+j,n*i+j-1) = -1;
            A(n*i+j,n*i+j+1) = -1; A(n*i+j,n*(i-1)+j) = -1;
            A(n*i+j,n*(i+1)+j) = -1;
        end
    end
end
```

Listing 2: LU_Factorization.m

```
clear
% L = [100,90,80;60,50,40];
% V = [90,70;80,60;70,50];
L = [linspace(120,71,50);linspace(69,44.5,50)];
V = [linspace(114.6,75.4,50);linspace(74,44.6,50)]';
[T,A,b] = data2equation(L,V);

tic;
%矩阵LU分解：A=LU,L为下三角矩阵，U为上三角矩阵。
%其中，L对角线元素为1.
n=size(A);n=n(1);L=eye(n);U=zeros(n);
U(1,1)=A(1,1)/L(1,1);
if L(1,1)*U(1,1)==0
    disp('Factorization impossible')
    return
end
U(1,2:n)=A(1,2:n)/L(1,1);
L(2:n,1)=A(2:n,1)/U(1,1);
for i=2:n-1
    U(i,i)=(A(i,i)-L(i,1:i-1)*U(1:i-1,i))/L(i,i);
    if L(i,i)*U(i,i)==0
        disp('Factorization impossible')
        return
    end
    U(i,i+1:n) = (A(i,i+1:n)-L(i,1:i-1)*U(1:i-1,i+1:n))/L(i,i);
    L(i+1:n,i) = (A(i+1:n,i)-L(i+1:n,1:i-1)*U(1:i-1,i))/U(i,i);
end
U(n,n) = (A(n,n)-L(n,1:n-1)*U(1:n-1,n))/L(n,n);
```

```
29
30 % 计算LUx=b
31 % 1、求z，使得Lz=b
32 z = zeros(n,1);
33 z(1) = b(1);
34 for i = 2:n
35     z(i) = b(i) - L(i,1:i-1)*z(1:i-1);
36 end
37 % 2、求x，使得Ux=z
38 x = zeros(n,1);
39 x(n) = z(n)/U(n,n);
40 for i = 1:n-1
41     x(n-i) = (z(n-i) - U(n-i,n-i+1:n)*x(n-i+1:n))/U(n-i,n-i);
42 end
43 toc;
44
45 %绘制诸点温度表
46 Tx = reshape(roundn(x,-2),size(T,2)-2,size(T,1)-2)';
47 T(2:(size(Tx,1)+1),2:(size(Tx,2)+1)) = Tx;
```

Listing 3: Jacobi_1.m

```
1  clear
2  % L = [100,90,80;60,50,40];
3  % V = [90,70;80,60;70,50];
4  L = [linspace(120,71,50);linspace(69,44.5,50)];
5  V = [linspace(114.6,75.4,50);linspace(74,44.6,50)]';
6  % L = [linspace(120,71,100);linspace(69,44.5,100)];
7  % V = [linspace(114.6,75.4,100);linspace(74,44.6,100)]';
8  [T,A,b] = data2equation(L,V); %用原始数据生成A和b
9
10 tic;
11 %A_0：定位矩阵A中的非零元素，仅针对非零元素进行迭代
12 n=size(A,1); A_0 = {};
13 for i = 1:n
14     a = find(A(i,:)≠0); a(find(a==i)) = [];
15     A_0{end+1} = a;
16 end
17 % Jacobi方法
18 der=10^-10; x0=zeros(n,1)+1; x=zeros(n,1);
19 k=1; N=100000;
20 while max(abs(x-x0)) > der
21     x0=x;
22     for i=1:n
23         %向量积简洁版
24         x(i) = (b(i)-A(i,A_0{i})*x0(A_0{i}))/A(i,i);
25     end
26     %下面的内容用于避免算法不收敛造成死循环
27     if k>N
28         disp('Maximum number of iterations exceeded')%此时算法可能不收敛
29         break
30     end
31     k=k+1;
32 end
33 toc;
34
```

```
%绘制诸点温度表
Tx = reshape(roundn(x,-2),size(T,2)-2,size(T,1)-2)';
T(2:(size(Tx,1)+1),2:(size(Tx,2)+1)) = Tx;
```

Listing 4: Jacobi_2.m

```
clear
% L = [100,90,80;60,50,40];
% V = [90,70;80,60;70,50];
L = [linspace(120,71,50);linspace(69,44.5,50)];
V = [linspace(114.6,75.4,50);linspace(74,44.6,50)]';
% L = [linspace(120,71,100);linspace(69,44.5,100)];
% V = [linspace(114.6,75.4,100);linspace(74,44.6,100)]';
[T,A,b] = data2equation(L,V); %用原始数据生成A和b

tic;
%A_0：定位矩阵A中的非零元素，仅针对非零元素进行迭代
n=size(A,1); A_0 = {};
for i = 1:n
    a = find(A(i,:)≠0); a(find(a==i)) = [];
    A_0{end+1} = a;
end
% Jacobi方法
der=10^-10; x0=zeros(n,1)+1; x=zeros(n,1);
k=1; N=100000;
while max(abs(x-x0)) > der
    x0=x;
    for i=1:n
        xi=b(i);
        for j = A_0{i}
            xi = xi - A(i,j)*x0(j);
        end
        x(i)=xi/A(i,i);
    end
    %下面的内容用于避免算法不收敛造成死循环
    if k>N
        disp('Maximum number of iterations exceeded')%此时算法可能不收敛
        break
    end
    k=k+1;
end
toc;

%绘制诸点温度表
Tx = reshape(roundn(x,-2),size(T,2)-2,size(T,1)-2)';
T(2:(size(Tx,1)+1),2:(size(Tx,2)+1)) = Tx;
```

Listing 5: GS_and_SOR_1.m

```
clear
% L = [100,90,80;60,50,40];
% V = [90,70;80,60;70,50];
L = [linspace(120,71,50);linspace(69,44.5,50)];
V = [linspace(114.6,75.4,50);linspace(74,44.6,50)]';
```

```
% L = [linspace(120,71,100);linspace(69,44.5,100)];
% V = [linspace(114.6,75.4,100);linspace(74,44.6,100)]';
[T,A,b] = data2equation(L,V); %用原始数据生成A和b

tic;
%A_0：定位矩阵A中的非零元素，仅针对非零元素进行迭代
n=size(A,1); A_0 = {};
for i = 1:n
    a = find(A(i,:)≠0); a(find(a==i)) = [];
    A_0{end+1} = a;
end
% Gauss-Seidel Method & SOR method
% 选择方法时，在对应位置（25行或26行）注释掉另一个方法的语句即可。
w=1.8; %SOR method
der=10^-10; x0=zeros(n,1)+1; x=zeros(n,1);
k=1; N=100000;
while max(abs(x-x0)) > der
    x0=x;x_d = x0;
    for i=1:n
        %向量积简洁版
        %x(i) = (b(i)-A(i,A_0{i})*x_d(A_0{i}))/A(i,i);%G-S方法
        x(i) = w*(b(i)-A(i,A_0{i})*x_d(A_0{i}))/A(i,i)+(1-w)*x0(i);%SOR方法
        x_d(i) = x(i);
    end
    %下面的内容用于避免算法不收敛造成死循环
    if k>N
        disp('Maximum number of iterations exceeded')%此时算法可能不收敛
        break
    end
    k=k+1;
end
toc;

%绘制诸点温度表
Tx = reshape(roundn(x,-2),size(T,2)-2,size(T,1)-2)';
T(2:(size(Tx,1)+1),2:(size(Tx,2)+1)) = Tx;
```

Listing 6: GS_and_SOR_2.m

```
clear
% L = [100,90,80;60,50,40];
% V = [90,70;80,60;70,50];
L = [linspace(120,71,50);linspace(69,44.5,50)];
V = [linspace(114.6,75.4,50);linspace(74,44.6,50)]';
% L = [linspace(120,71,100);linspace(69,44.5,100)];
% V = [linspace(114.6,75.4,100);linspace(74,44.6,100)]';
[T,A,b] = data2equation(L,V); %用原始数据生成A和b

tic;
%A_0：定位矩阵A中的非零元素，仅针对非零元素进行迭代
n=size(A,1); A_0 = {};
for i = 1:n
    a = find(A(i,:)≠0); a(find(a==i)) = [];
    A_0{end+1} = a;
end
% Gauss-Seidel Method & SOR method
```

```
% 选择方法时，在对应位置（32行或33行）注释掉另一个方法的语句即可。
w=1.8; %SOR method
der=10^-10; x0=zeros(n,1)+1; x=zeros(n,1);
k=1; N=100000;
while max(abs(x-x0)) > der
    x0=x;
    for i=1:n
        xi=b(i);
        for j = A_0{i}
            if j < i
                xi = xi - A(i,j)*x(j);
            elseif j > i
                xi = xi - A(i,j)*x0(j);
            end
        end
        %x(i)=xi/A(i,i); %Gauss-Seidel 方法
        x(i)=w*xi/A(i,i)+(1-w)*x0(i);  % SOR 方法
    end
    %下面的内容用于避免算法不收敛造成死循环
    if k>N
        disp('Maximum number of iterations exceeded')
        break
    end
    k=k+1;
end
toc;

%绘制诸点温度表
Tx = reshape(roundn(x,-2),size(T,2)-2,size(T,1)-2)';
T(2:(size(Tx,1)+1),2:(size(Tx,2)+1)) = Tx;
```