应用数学导论大作业

敖睿成

1 摘要

考虑方程

$$\begin{cases}\dfrac{\partial u}{\partial t}=\Delta u\\ \text{边值条件}\end{cases}$$

本次实验中，依次针对一维情形，$\Omega=[0,1]\times[0,1]$，L-型区域上的泊松问题，分别使用有限元方法，和中心差分方法求解方程, 并给出相关理论分析和实验结果。

2 一维自适应有限元方法

对一维问题

$$\begin{cases}-u''=f \qquad \Omega=(0,L)\\ u'(0)=\kappa_0(u(0)-g_0)\\ u'(L)=\kappa_L(u(L)-g_L)\end{cases}$$

对于 $N+1$ 个结点 $x_0=0,x_1=\frac{L}{N},...x_N=L$, 在子区间 $[x_{i-1},x_{i+1}](i=1,2,...,N-1)$ 上取分段线性函数:

$$\phi_i(x)=\begin{cases}\dfrac{x-x_{i-1}}{h_i}, & x\in[x_{i-1},x_i]\\ \dfrac{x_{i+1}-x}{h_{i+1}}, & x\in[x_i,x_{i+1}]\\ 0, & x\notin[x_{i-1},x_{i+1}]\end{cases}$$

其中 $h_i=x_i-x_{i-1}$, 对于边界两个结点, 考虑 $x_0,x_1;x_{N-1},x_N$ 对应的两个两点线性插值函数，这样得到 $N+1$ 个基函数 $\phi_0,\phi_1,...,\phi_N$, 现求解 $v(x)=\sum_{j=0}^{N}u_j\phi_j(x)$, 使得

$$-\int_0^1 u''(x)v(x)dx=\int_0^1 f(x)v(x)dx$$

成立，通过变分方法得到:

$$\begin{cases}-\dfrac{u_{i-1}}{h_i}+(\dfrac{1}{h_i}+\dfrac{1}{h_{i+1}})u_i-\dfrac{u_{i+1}}{h_{i+1}}=\displaystyle\int_{x_{i-1}}^{x_{i+1}}f\phi'(x)dx \qquad i=1,2,...,N-1\\ (\dfrac{1}{h_1}+\kappa_0)u_0-\dfrac{u_1}{h_1}=f_0\\ (\dfrac{1}{h_N}+\kappa_L)u_N-\dfrac{u_{N-1}}{h_N}=f_N\end{cases}$$

令 $a_i = \frac{1}{h_i} + \frac{1}{h_{i+1}}, i = 1, 2, ..., N-1,$

$$A = \begin{pmatrix} \frac{1}{h_1} + \kappa_0 & -\frac{1}{h_1} & & & \\ -\frac{1}{h_1} & a_1 & -\frac{1}{h_2} & & \\ & -\frac{1}{h_2} & a_2 & \ddots & \\ & & \ddots & \ddots & -\frac{1}{h_N} \\ & & & -\frac{1}{h_N} & \frac{1}{h_N} + \kappa_L \end{pmatrix}$$

得到方程

$$Au = F$$

分别应用均匀剖分和自适应方法，在总点数 $N = 10, 20, 80$ 时得到如下结果:
$\kappa_0 = 10^6, k_1 = 0, g_0 = 0, f(x) = e^{-100(x-0.5)^2}$ 时，

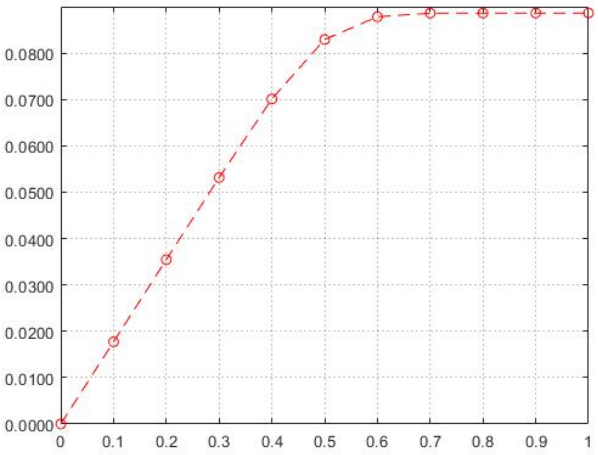

图 1: 均匀剖分 10

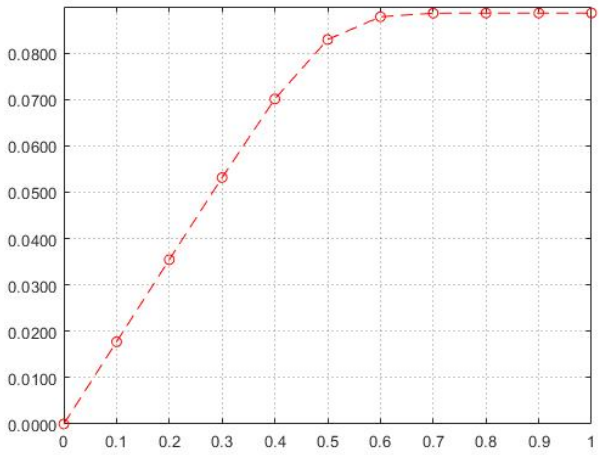

图 2: 自适应 10

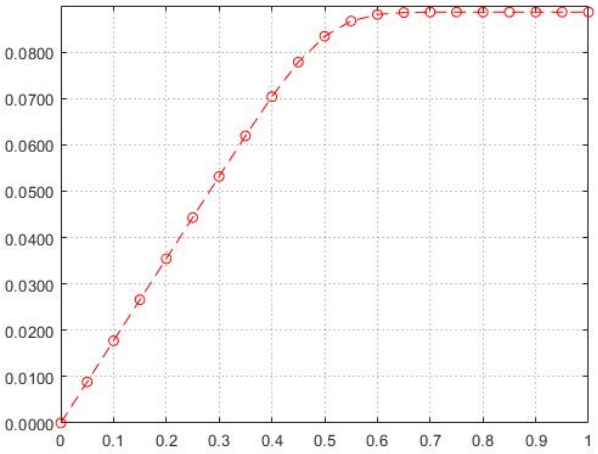

图 3: 均匀剖分 20

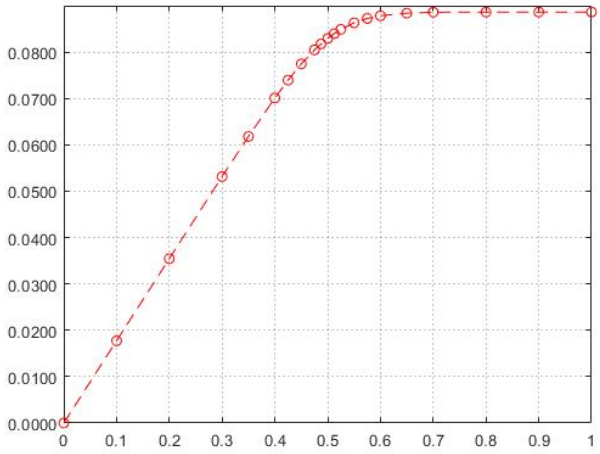

图 4: 自适应 20

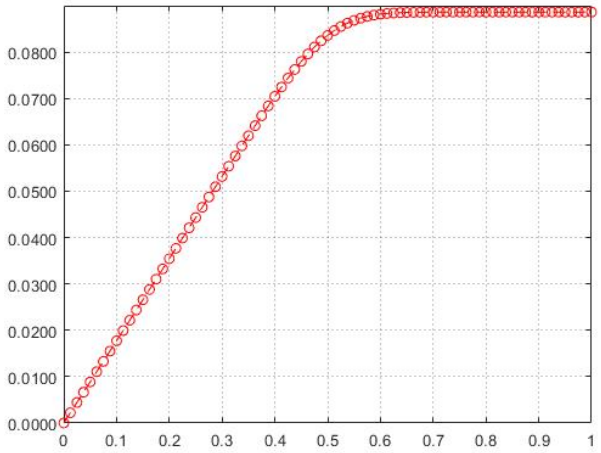

图 5: 均匀剖分 80

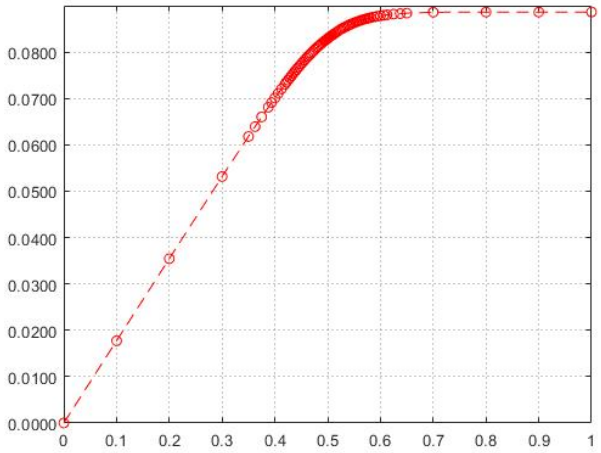

图 6: 自适应 80

$\kappa_0 = 10^6, k_1 = 10^5, g_0 = 0, g_L = 0, f(x) = e^{-100(x-0.5)^2}$ 时,

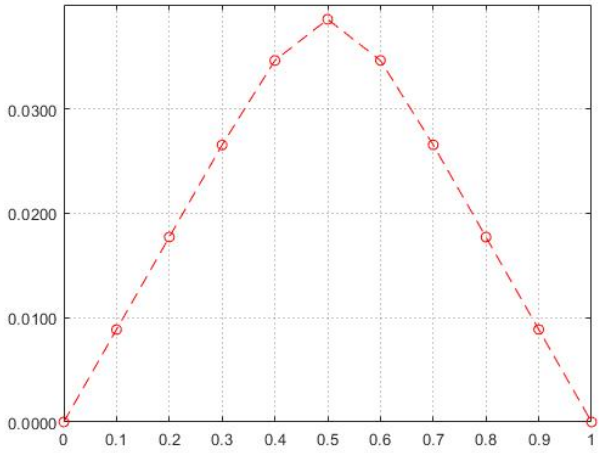

图 7: 均匀剖分 10

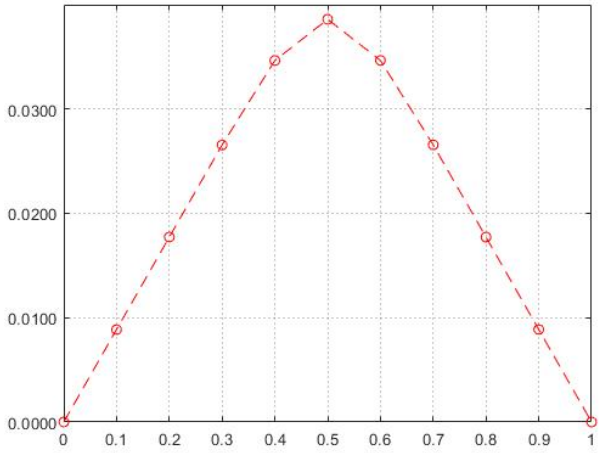

图 8: 自适应 10

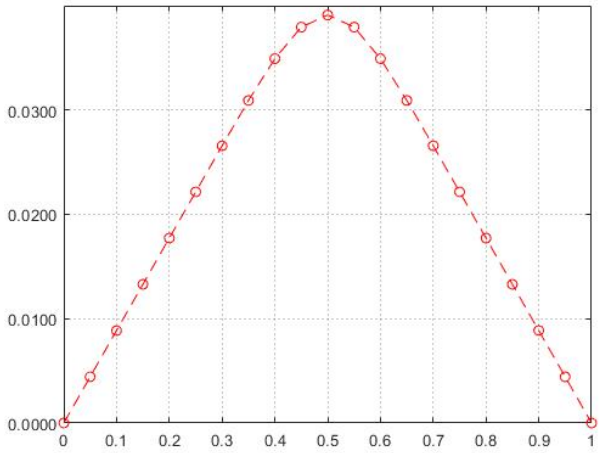

图 9: 均匀剖分 20

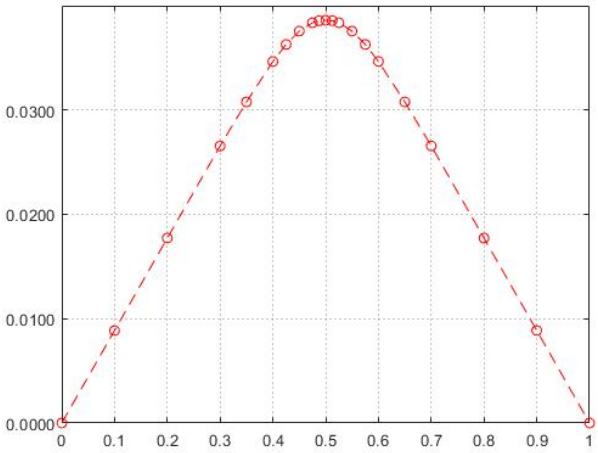

图 10: 自适应 20

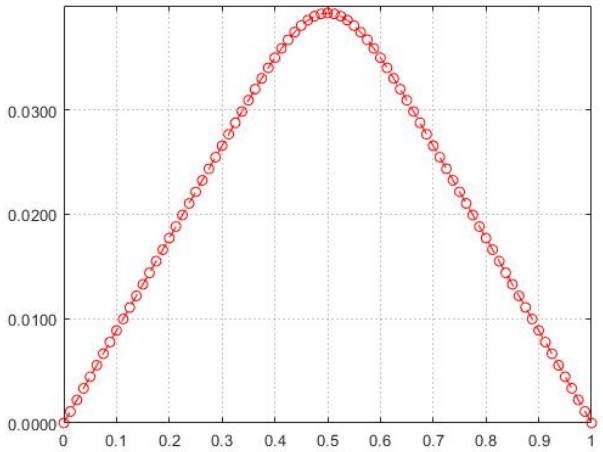

图 11: 均匀剖分 80

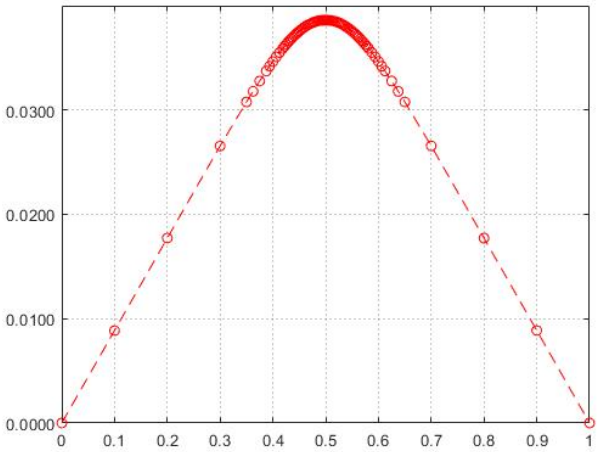

图 12: 自适应 80

$\kappa_0 = 10^6, k_1 = 0, g_0 = -1, f(x) = e^{-100(x-0.5)^2}$ 时,

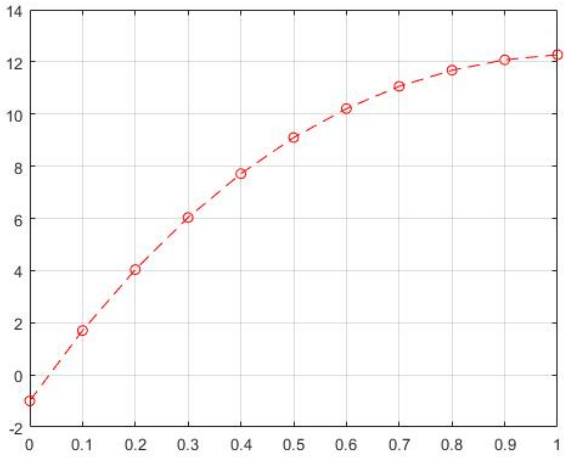

图 13: 均匀剖分 10

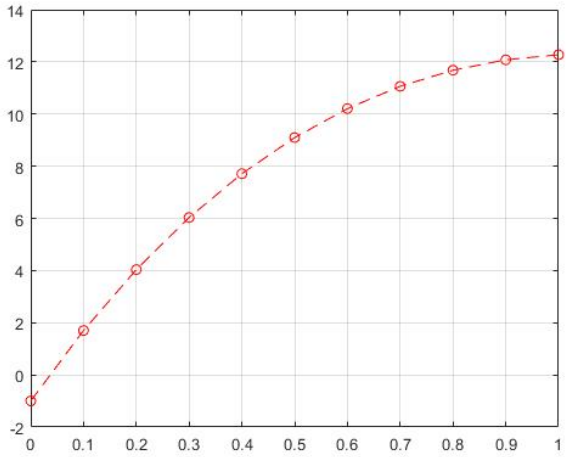

图 14: 自适应 10

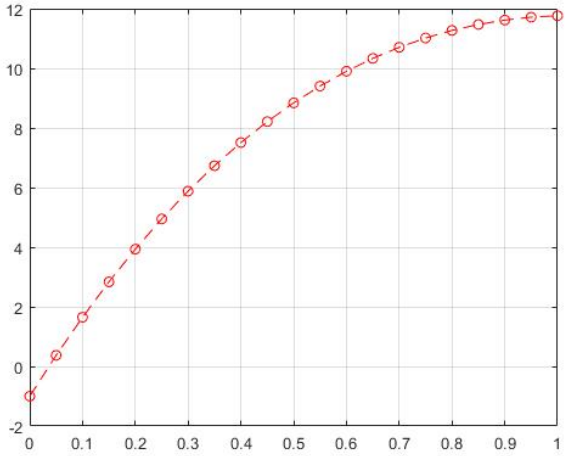

图 15: 均匀剖分 20

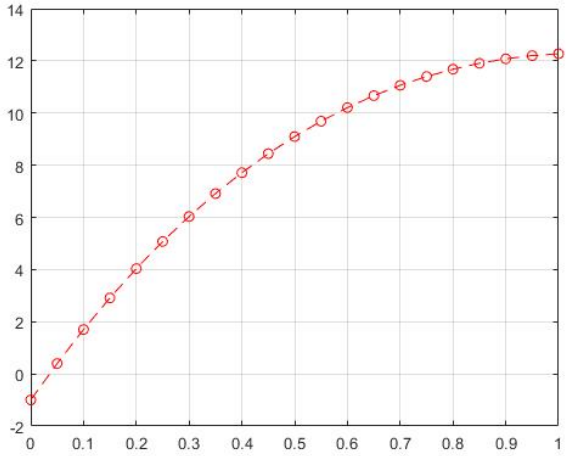

图 16: 自适应 20

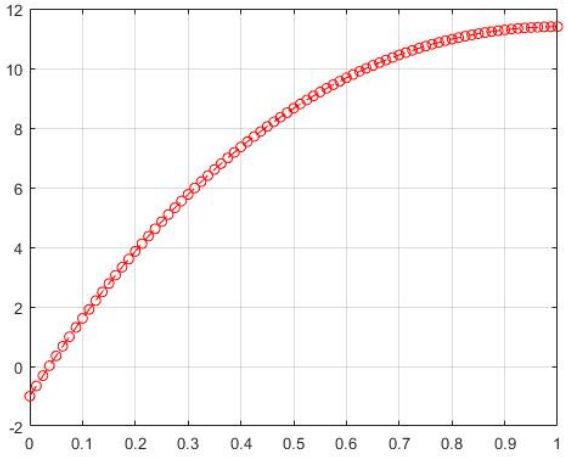

图 17: 均匀剖分 80

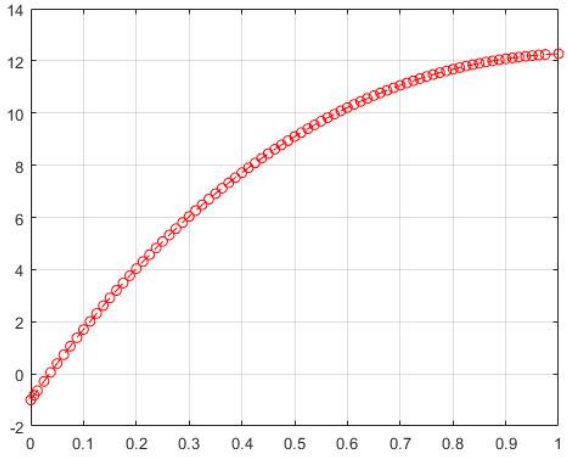

图 18: 自适应 80

可以看到，相比较均匀剖分，自适应方法在曲率大的点附近加密得较细，所得函数也更为平滑。

3 正方形区域泊松方程

3.1 问题

考虑热传导方程:

$$\begin{cases} \dfrac{\partial u}{\partial t} = \Delta u \\ u|_{\partial\Omega\times[0,1]} = 0, \Omega = [0,1]\times[0,1] \\ u|_{t=0} = \sin(\pi x)\sin(\pi y) \end{cases}$$

它有解析解 $u = e^{-2\pi^2 t}\sin(\pi x)\sin(\pi y)$，下面我们使用不同数值方法计算方程近似解，并给出相关分析。

3.2 稳定性分析

考察 Laplace 算子

$$\Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$$

我们使用中心差分方法

$$L_{h_x,h_y}U_{i,j}^m \xlongequal{\text{def}} \frac{U_{i-1,j}^m - 2U_{i,j}^m + U_{i+1,j}^m}{h_x^2} + \frac{U_{i,j-1}^m - 2U_{i,j}^m + U_{i,j+1}^m}{h_y^2}$$

这里 h_x, h_y 为对应分量的区间步长，对于 $\frac{\partial u}{\partial t}$，我们使用一阶向前差分方法:

$$D_k U_{i,j}^m \xlongequal{\text{def}} \frac{U_{i,j}^{m+1} - U_{i,j}^m}{k}$$

这里 k 为时间步长。现在考虑等式:

$$(1-\theta)L_{h_x,h_y}U_{i,j}^m + \theta L_{h_x,h_y}U_{i,j}^{m+1} = D_k U_{i,j}^m$$

当 $\theta = 1$ 时，为隐式格式，$\theta = 0.5$ 时，为 Crank-Nicolson 格式，$\theta = 0$ 时，为显式格式。令

$$\tilde{U} = u\left(ih_x, jh_y, \left(m+\frac{1}{2}\right)k\right)$$

应用 Taylor 公式，可以得到

$$\begin{aligned} L_{hx,hy}U_{i,j}^m =& \tilde{U}_{xx} + \tilde{U}_{yy} + \frac{2}{3!}\left(3\tilde{U}_{txx}\left(-\frac{1}{2}k\right) + 3\tilde{U}_{tyy}\left(-\frac{1}{2}k\right)\right) \\ &+ \frac{2}{4!}\left(\tilde{U}_{xxxx}h_x^2 + \tilde{U}_{yyyy}h_y^2\right) + O(k^2 + h_x^4 + h_y^4) \end{aligned}$$

$$\begin{aligned} L_{hx,hy}U_{i,j}^{m+1} =& \tilde{U}_{xx} + \tilde{U}_{yy} + \frac{2}{3!}\left(3\tilde{U}_{txx}\frac{1}{2}k + 3\tilde{U}_{tyy}\frac{1}{2}k\right) \\ &+ \frac{2}{4!}\left(\tilde{U}_{xxxx}h_x^2 + \tilde{U}_{yyyy}h_y^2\right) + O(k^2 + h_x^4 + h_y^4) \end{aligned}$$

从而有:

$$(1-\theta)L_{h_x,h_y}U_{i,j}^m+\theta L_{h_x,h_y}U_{i,j}^{m+1}-\Delta\tilde{U}=\left(\left(\theta-\frac{1}{2}\right)k+\frac{h_x^2}{12}\right)\tilde{U}_{xxxx}+\left(\left(\theta-\frac{1}{2}\right)k+\frac{h_y^2}{12}\right)\tilde{U}_{yyyy}$$
$$+(2\theta-1)\,k\tilde{U}_{xxyy}+O(k^2+h_x^4+h_y^4)$$

这样截断误差 $\mathcal{E}$ 满足

$$\mathcal{E}=\begin{cases}O(k^2+h_x^2+h_y^2) & \theta=0.5\\ O(k+h_x^2+h_y^2) & \theta\neq 0.5\\ O(k+h_x^4+h_y^4) & h_x=h_y,\theta=0.5-1/12\lambda\end{cases}$$

可以看出，C-N 方法具有较高的截断误差阶数，现在考虑 Fourier 函数

$$U_{j,k}^m=\lambda_\alpha^m e^{i(\alpha_x x_j+\alpha_y y_k)},\quad \alpha=(\alpha_x,\alpha_y)$$

解得

$$\lambda_k=\frac{1-4(1-\theta)\left(\lambda_x\sin^2(\alpha_x h_x/2)+\lambda_y\sin^2(\alpha_y h_y/2)\right)}{1+4\theta\left(\lambda_x\sin^2(\alpha_x h_x/2)+\lambda_y\sin^2(\alpha_y h_y/2)\right)}$$

因此，我们有稳定性条件

$$\begin{cases}2(\lambda_x+\lambda_y)(1-2\theta)\leq 1 & 0\leq\theta<1/2\\ \text{无条件收敛} & 1/2\leq\theta\leq 1\end{cases}$$

这表明当 $h_x=h_y=h$ 时，对于显式格式，我们需要选取 $\lambda\leq\frac{1}{4}$, 即 $h^2\geq 4k$，才能得到收敛结果。

3.3 数值实验

考虑由中间 $(N-1)\times(N-1)$ 个点，其中 $N=\frac{1}{h}$ 为单元网格长，则对应的差分矩阵

$$L_h=\begin{pmatrix}A_h & I_{N-1}/h & & & \\ I_{N-1}/h^2 & A_h & I_{N-1}/h^2 & & \\ & I_{N-1}/h^2 & A_h & \ddots & \\ & & \ddots & \ddots & I_{N-1}/h^2\\ & & & I_{N-1}/h^2 & A_h\end{pmatrix}$$

$$A_h=\begin{pmatrix}-2/h^2-2/h^2 & 1/h^2 & & & \\ 1/h^2 & -2/h^2-2/h^2 & 1/h^2 & & \\ & 1/h^2 & -2/h^2-2/h^2 & \ddots & \\ & & \ddots & \ddots & 1/h^2\\ & & & 1/h^2 & -2/h^2-2/h^2\end{pmatrix}$$

这样，得到如下方程

$$(I-k\theta L_h)U^{m+1}=(I+k(1-\theta)L_h)U^m$$

其中 U 为对应的网格向量化。下面考虑几种求解对应方程的方法。

Cholesky 分解

考察方程 $Ax = b$, 其中 A 为对称正定矩阵，则我们有如下 Cholesky 分解用以求解方程：

Algorithm 1: 利用向量外积的 cholesky 分解

Input: $A \in \mathbb{R}^{n\times n}$

Output: $L \in \mathbb{R}^{n\times n}$, $LL^{\mathrm{T}} = A$

for $j = 1 : n$ **do**
  if $j > 1$ **then**
    $A(j:n, j) = A(j:n, j) - A(j:n, 1:j-1)A(j, 1:j-1)^{\mathrm{T}}$
  $A(j:n, j) = A(j:n, j)/\sqrt{A(j, j)}$
return $tril(A)$;

计算量约为 $O(n^3/3)$, 是直接进行 Gauss 消元法的一半，在矩阵阶数较小时具有很快速度，但当矩阵阶数较大时，可以使用分块的方法加快速度。

Gauss-Seidel 迭代法

令 $A = D - L - U$, 其中 D, L, U 分别为对角矩阵、下三角矩阵和上三角矩阵，则我们有如下算法：

Algorithm 2: G-S 迭代法

Input: $A \in \mathbb{R}^{n\times n}$,$b \in \mathbb{R}^n$, **TOLERANCE** tol, **INITIALVALUE** x_0 **MAXITERATION** ite

Output: $x \in \mathbb{R}^n$, $Ax \approx b$

Divide A into D,L and U
$res = res_0 = b - Ax$
while $norm(res)/norm(res_0) > tol$ ***or*** $ite_{number} \leq ite$ **do**
  Update $y \leftarrow Ux$
  Get x by solving $(D - L)x = y + b$
  Update $res = -y$
  Update $Ux \leftarrow res + y$
  if $norm(res) \leq tol * norm(res_0)$ **then**
    return x
return x;

这里利用到了 $\mathrm{res} = b - Ax^{m+1} = b - (D - L)x^{m+1} + Ux^{m+1} = Ux^{m+1} - Ux^m$。可以证明，当 A 为对称正定矩阵时，G-S 迭代法是收敛的。

多重网格法

对于单元格长为 $h = 1/N$ 的细网格，我们希望找到一个较好的初始值，从而加快迭代法收敛的速度，故考虑在细网格上先进行若干次迭代，将误差限制在粗网格上，在粗网格解关于误差的方程，再把近似解提升回细网格，这样两者相加，可以认为得到了更近的初值，再重复这样的操作，直到误差满足要求，这里可以重复迭代，算法如下：

Algorithm 3: 多重网格法

Input: $A \in \mathbb{R}^{n\times n}$,$b \in \mathbb{R}^n$, **EDGE** h, **MAXITERATION** ite_1, **INITIALVALUE** x_0

Output: $x \in \mathbb{R}^n$, $Ax \approx b$

if $size(x) < threshold$ **then**
  Solve $Ax = b$ by **G-S** with **INITIALVALUE** x_0
else
  Solve $Au = b$ by **G-S** with **INITIALVALUE** x_0 and **MAXITERATION** ite_1
  Get residue: $res \leftarrow b - Au$
  Get coarse residue: $\widehat{res} \leftarrow I_h^{2h} res$
  $\widehat{A} \leftarrow I_h^{2h} A\, I_{2h}^h$
  Solve $\widehat{A}\hat{e} = \widehat{res}$ with **EDGE** $2 * h$, **INITIALVALUE 0** and **MAXITERATION** ite by **G-S**
  $e \leftarrow I_{2h}^h \hat{e}$
  Update $u \leftarrow u + e$
  Solve $Ax = b$ by **G-S** with **INITIALVALUE** u and **MAXITERATION** ite_2
return x;

这里 I_h^{2h}, I_{2h}^h 分别为限制和提升矩阵。当初始值较差时，多重网格的速度一般比直接 G-S 迭代快。

数值结果

对于近似解 $\tilde{U}_h$, 在每个单元上使用双线性函数来逼近原区域。对于单元 K，定义映射:

$$\tilde{F} : \xi = \frac{2(x - x_c)}{h_x}, \eta = \frac{2(y - y_c)}{h_y}, \qquad (x, y) \in K, (\xi, \eta) \in \tilde{K}$$

定义如下的基函数:

$$\Phi_1 = \frac{(1-\xi)(1-\eta)}{4}, \Phi_2 = \frac{(1-\xi)(1+\eta)}{4}$$
$$\Phi_3 = \frac{(1+\xi)(1-\eta)}{4}, \Phi_4 = \frac{(1+\xi)(1+\eta)}{4}$$

令 $u_h(x, y)|_K = \hat{u}(\xi, \eta) = \sum_{i=1}^4 u_i \Phi_i(\xi, \eta)$, 我们计算误差

$$||u(x, y, 1) - u_h(x, y, 1)||_0 = (\int_\Omega (u(x, y, 1) - u_h(x, y, 1))^2 dxdy)^{1/2}$$

和相对误差

$$||u(x, y, 1) - u_h(x, y, 1)||_0 / |\int_\Omega u dxdy|$$

数值结果如下:

空间步长 h	时间步长 k	μ	相对误差		
			显式格式	隐式格式	Crank-Nicolson 格式
1/64	1/256	16	$+\infty$	2.6179	4.5217e-2
	1/512	8	$+\infty$	0.9134	7.5452e-3
	1/2048	2	$+\infty$	0.2626	5.3124e-3
	1/4096	1	$+\infty$	7.6823e-2	5.2517e-3
	1/16384	1/4	3.1562e-2	3.2415e-2	5.4135e-3
1/128	1/1024	16	$+\infty$	5.3681	5.3268e-3
	1/4096	4	$+\infty$	0.4588	3.6782e-3
	1/16384	1	$+\infty$	7.6582e-2	2.5796e-3
	1/65536	1/4	8.3725e-3	8.2373e-3	2.7599e-3

表 1: 不同离散格式在最终时间层 $t = 1$ 的稳定性比较

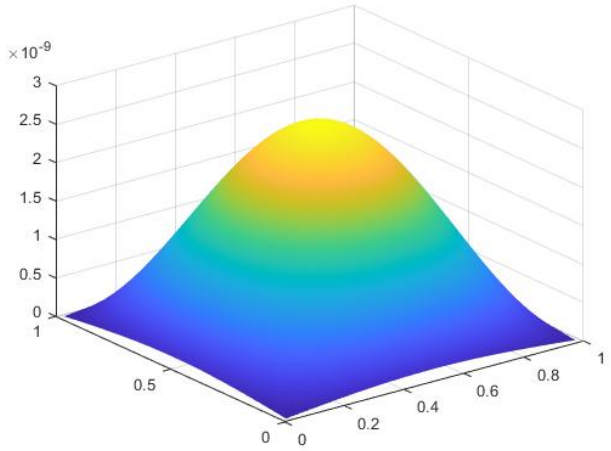

图 19: $t = 1$ 时图像

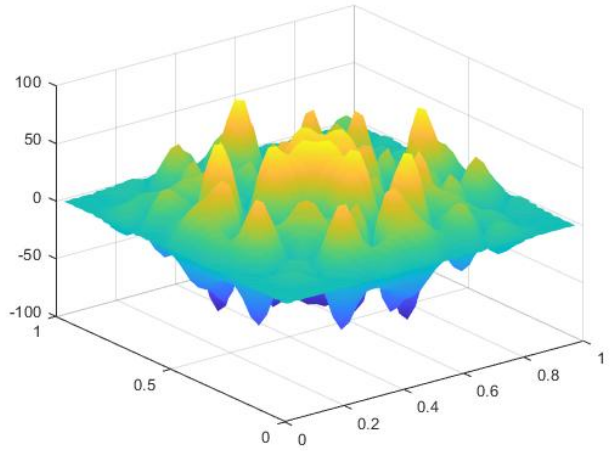

图 20: $\mu > 1/4$ 时显式格式

可以看到，与理论分析相同，当网格比 $\mu > \frac{1}{4}$ 时，显式格式是不稳定的，而 C-N 方法在网格比较大时就有很好的稳定性，但随着网格比和步长的减小，三者的表现最终接近。

现在取 $h = 1/32, 1/128, 1/512, k = 1/512$，分别应用 Cholesky 分解，G-S 迭代法和 V-cycle 多重网格法求解 $t = 1$ 的解，对于迭代法，每次以上一个时间层的解为初始值，所得时间分别如下：

时间步长 k	空间步长 h	总时间 (s)		
		多重网格 V	Cholesky	Gauss-Seidel
1/512	1/32	1.3298e+1	7.2836	7.0681
	1/128	3.6337e+2	1.0274e+2	2.3415e+3
	1/512	6.3446e+3	7.5289e+3	\

表 2: 时间步长 k=1/512 时，三种方法求方程在 $t = 1$ 解的总时间

可以看出多重网格的时间大致为 $O(N^2)$。

对每个固定的 h, 分别用三种格式求近似解，对每一种格式，令 k 从大到小依次取 2 的幂，比较取不同 k 时求解的误差，取出其中最小的，并且计算误差

$$||u(x,y,1) - u_h(x,y,1)||_0$$

所得结果如下:

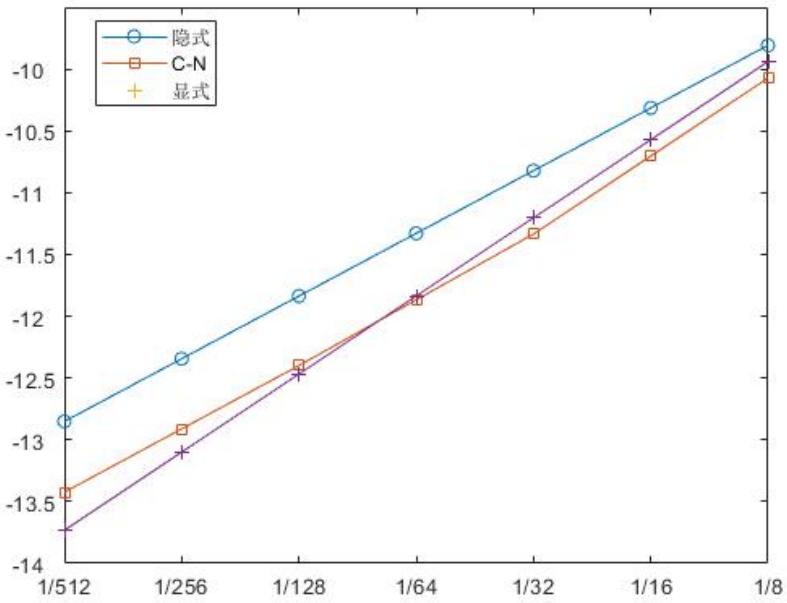

图 21: 对不同的 h，选取最佳步长所得误差

可以发现，虽然显式格式在 $\lambda > 1/4$ 时是发散的，但在适当的网格比下，反而可以得到很好的结果。

4 L 型区域上的泊松方程

4.1 问题

考虑如下泊松问题:

$$\begin{cases} -\Delta = f & (x,y) \in \Omega, \\ u|_{\partial\Omega} = 0 \end{cases}$$

其中 Ω 为如下 L 型区域，$u(x,y) = r^{\frac{2}{3}} sin(\frac{2}{3}\theta)(1-x^2)(1-y^2)$。

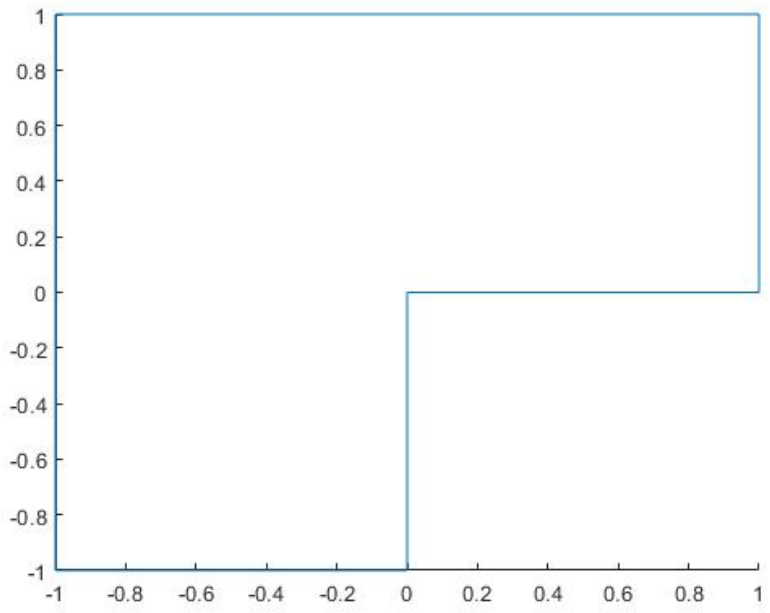

图 22: L 型区域

它的函数值和 Δ 在 $[0,1]\times[0,1]$ 区间上的图像如下:

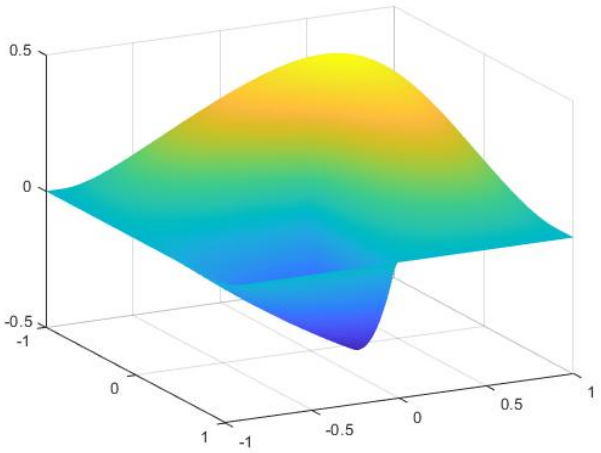

图 23: 函数值

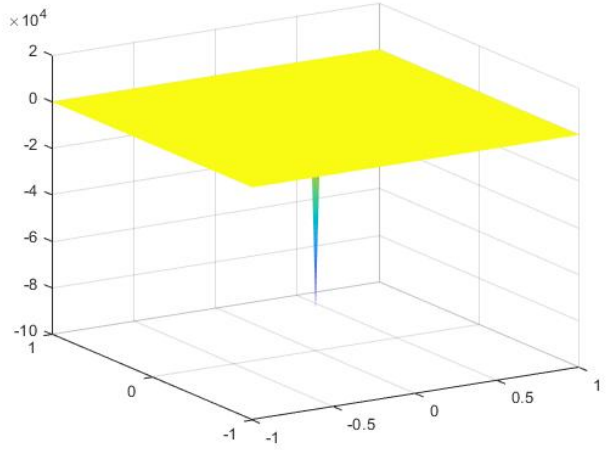

图 24: Δ

可以看到，在 $(0,0)$ 的邻域内 $\Delta\to+\infty$。下面我们分别使用均匀剖分和自适应剖分，利用差分方法求方程的近似解。

4.2 数值实验

LDL 分解

考察方程 $Ax=b$, 对于对称非正定矩阵 A,Cholesky 分解不再可行，此时可以考虑同样利用了对称性的 LDL

分解，得到 $A = LDL^{\mathrm{T}}$，依次求解 $Lz = b, Dy = z, L^{\mathrm{T}}x = y$ 得到方程的解，这三个方程分别为下三角、对角、上三角的，可以在 $O(n^2)$ 时间内求解，故 LDL 分解所需要的运算时间约为 $O(n^3/3)$ 的, 下面给出算法:

Algorithm 4: 利用向量外积的 LDL 分解

Input: $A \in \mathbb{R}^{n\times n}$

Output: $L, D \in \mathbb{R}^{n\times n}$, $LDL^{\mathrm{T}} = A$

for $j = 1 : n-1$ **do**
  $A(j+1:n, j) = A(j+1:n, j)/A(j, j)$
  $A(j+1:n, j+1:n) = A(j+1:n, j+1:n) - A(j+1:n, j) * A(j, j+1:n)$
$L = tril(A, -1); D = diag(A)$
return L, D;

当矩阵较大时，可以采用分块 LDL 分解, 算法如下:

Algorithm 5: 分块 LDL 分解

Input: $A \in \mathbb{R}^{n\times n}$, **MINISIZE**,**NUM**

Output: $L, D \in \mathbb{R}^{n\times n}$, $LDL^{\mathrm{T}} = A$

if $n \le$ ***MINISIZE*** **then**
  Solve $A = LDL^{\mathrm{T}}$ with **LDL**
subsize = **ceil**$(n/$**NUM**$)$
for $j = 1 : subsize : n$ **do**
  if j+$subsize > n$ **then**
    Solve $A(j:n, j:n) = LDL^{\mathrm{T}}$ with **algorithm 5**, update $tril(A(j:n, j:n))$ with L, D
  else
    Solve $A(j:j$+subsize$-1, j:j$+subsize$-1) = LDL^{\mathrm{T}}$ with **algorithm 5**
    Update $tril(A(j:j$+subsize$-1, j:j$+subsize$-1))$ with L, D
    Solve $LV = A(j:j$+subsize$-1, j + subsize : n)$
    Solve $DU^{\mathrm{T}} = V$
    Update $A(j$+subsize: $n, j:j$+subsize$-1) \leftarrow U$
    Update $A(j$+subsize: n, j+subsize: $n) \leftarrow A(j$+subsize: n, j+subsize: $n) - UV$
$L = tril(A, -1), D = diag(A)$
return L, D;

数值结果

将 L 型区域分别以步长 $h = h_x = h_y = 2/N, N = 32, 64, 128, 256, 512$ 进行均匀剖分，使用 LDL 分解，G-S 迭代，松弛因子为 1.5 的超松弛迭代法，V-cycle 多重网格法求解差分离散方程 $A_h U_h = F_h$ 的解 U_h 或近似解 $\tilde{U}_h$，其中系数矩阵为稀疏的对称正定矩阵，形如下图:

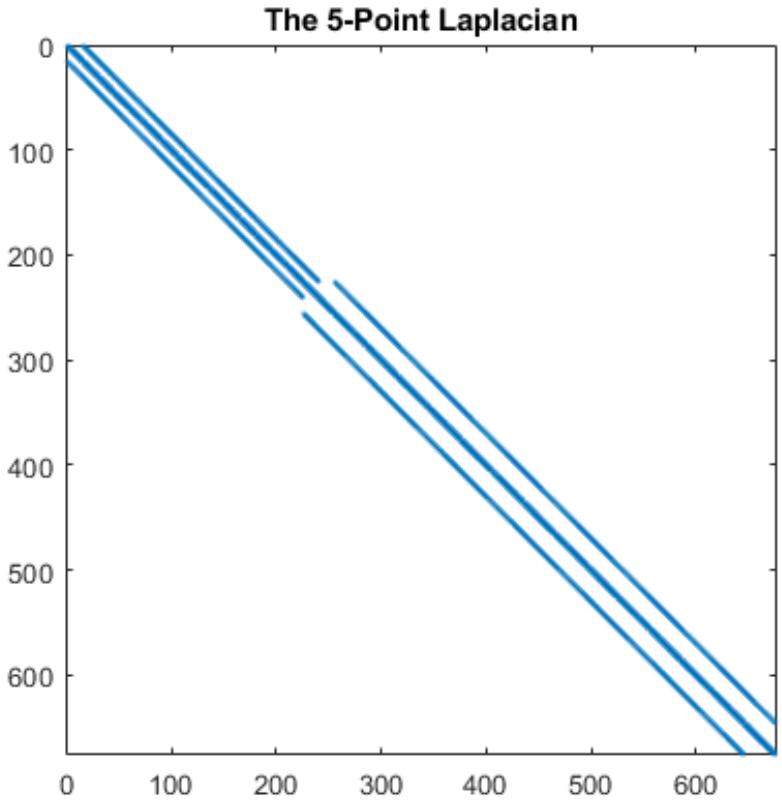

图 25: L 型差分系数矩阵

编号的顺序是从右上角 $(1-h, 1-h)$ 开始，从上到下，从右往左编号。迭代法的初始值为全零网格，近似解 $\tilde{U}_h$ 满足误差关系

$$||A_h\tilde{U}_h - F_h||_2/||F_h||_2 \leq 10^{-8}$$

所花的时间如下:

$N = 2/h$	总时间 (s)			
	LDL 分解	G-S 迭代 V	超松弛迭代	多重网格
32	5.1174e-2	1.3309e-2	8.5721e-3	3.4924e-3
64	2.9712e-1	2.9704	7.6794e-2	1.5284e-2
128	1.3829	6.1109	2.2015	6.6295e-2
256	1.7325e+1	1.0036e+2	3.6239e+1	2.6981e-1
512	1.9108e+2	2.1589e+3	7.5086e+2	1.0992

表 3: 时间步长 k=1/512 时，三种方法求方程在 $t = 1$ 解的总时间

接下来使用自适应剖分差分方法求解方程。格式如下:

Algorithm 6: L 型区域自适应方法

Input: INITIALGRID U_0, **ERROR** Γ_0,θ,**TOLERANCE** ϵ,$k=0$

Output: U, N

Update $\eta(\Gamma_{k-1}) \leftarrow \eta(\Gamma_k)$

while ***TRUE*** **do**

  if $\eta(\Gamma_k) \leq \epsilon$ **then**

    $N = k, U = U_k$ **Return**

  Find minimal subset unit set $\mathcal{M}_k$ such that $\eta(\mathcal{M}_k)^2 \geq \theta\eta(\Gamma_k)^2$

  Update $\Gamma_k \leftarrow \Gamma_{k+1}$

  Update $k \leftarrow k+1$

return N, U;

这里 $\eta(\Gamma)$ 为后验误差估计因子:

$$\eta_K^2 = h_K^2||f||_{L^2(K)}^2 + \sum_{e\in\mathcal{E}_K} h_e|||[\frac{\partial u_h}{\partial n_e}]||_{L^2(e)'}^2$$

其中 h_K 是单元 K 最长边的长度，$\mathcal{E}$ 是 K 所有不在边界的边的集合，h_e 是边 e 的长度，n_e 是 e 上的外法向方向，$[\frac{\partial u_h}{\partial n_e}]$ 是法向导数跨过 e 的跳跃，即

$$\frac{\partial u_h}{\partial n_e}|_{K^+} - \frac{\partial u_h}{\partial n_e}|_{K^-}$$

$\eta(G)$ 表示单元集合 G 中所有单元的后验误差之和。每次剖分把所选集合中的单元加细，并将边界上有两个悬点的单元也加细，使得任意一个单元的边界上只有至多一个悬点。需要注意的是，在计算中悬点取边两端点函数值的均值。分别以 $\theta = 0.8, 0.6, 0.3, 0.2, 0.1, \epsilon = 10^{-6}$, 一般来说，$\theta$ 越大，每次加密得网格越多，单步时间越长，收敛到精确解所需的步数越少。得出网格剖分图如下:

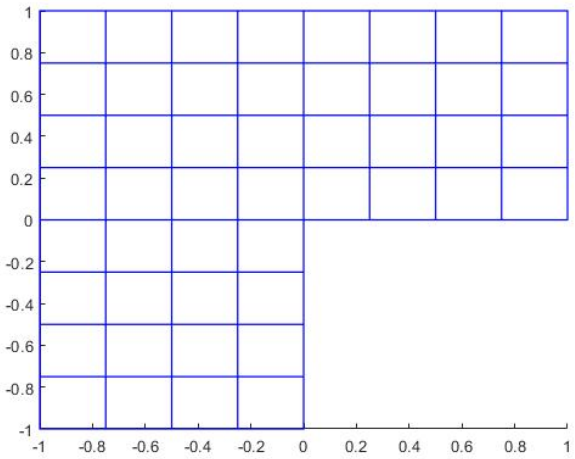

图 26: $\theta = 1$,1 次剖分

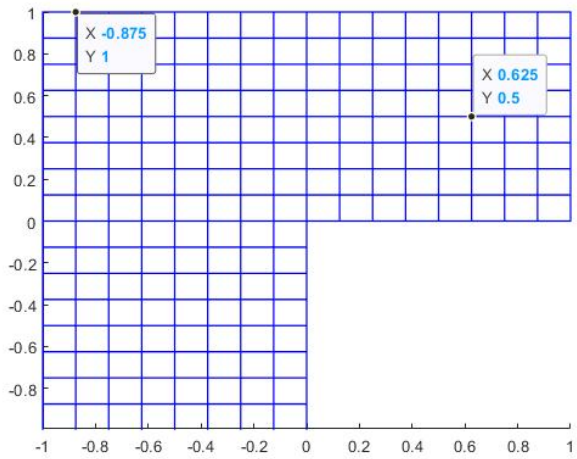

图 27: $\theta = 1$,2 次剖分

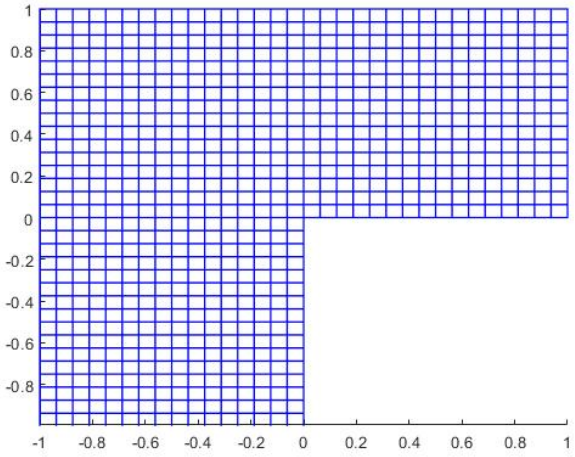

图 28: $\theta = 1$,3 次剖分

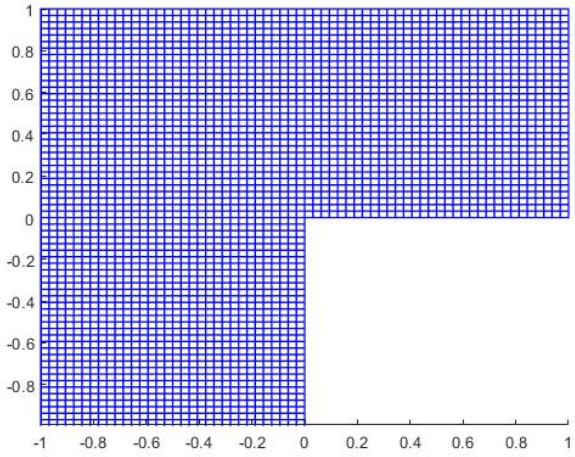

图 29: $\theta = 1$,4 次剖分

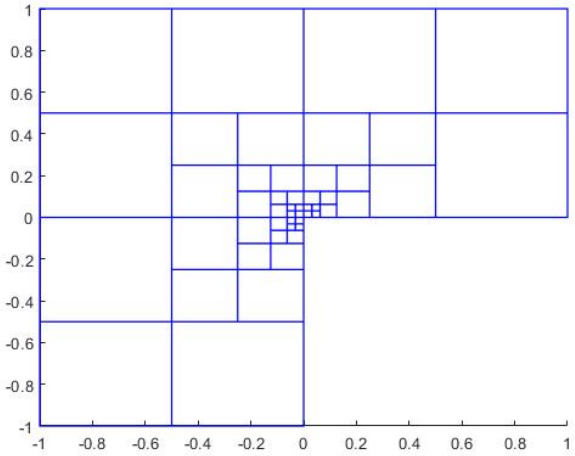

图 30: $\theta = 0.8$,4 次剖分

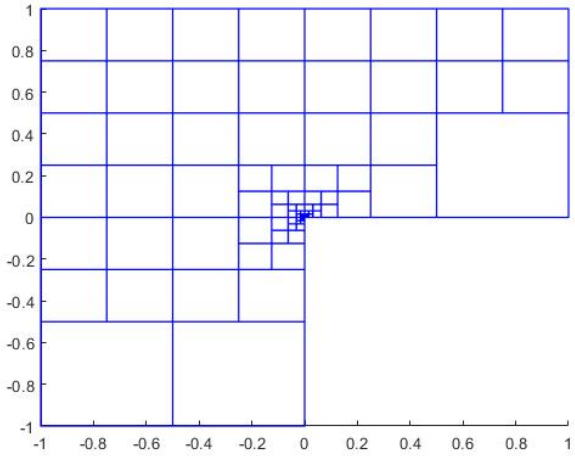

图 31: $\theta = 0.8$,7 次剖分

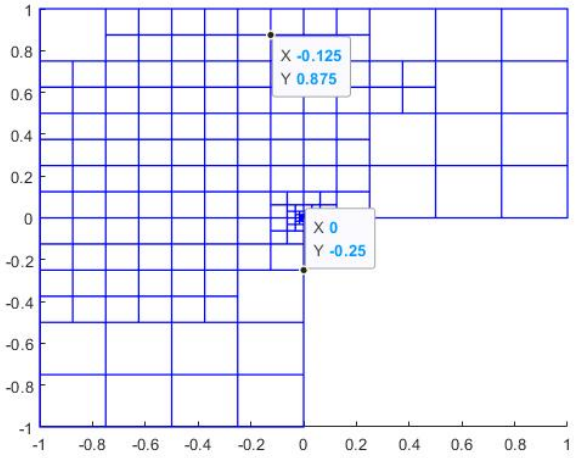

图 32: $\theta = 0.8$,8 次剖分

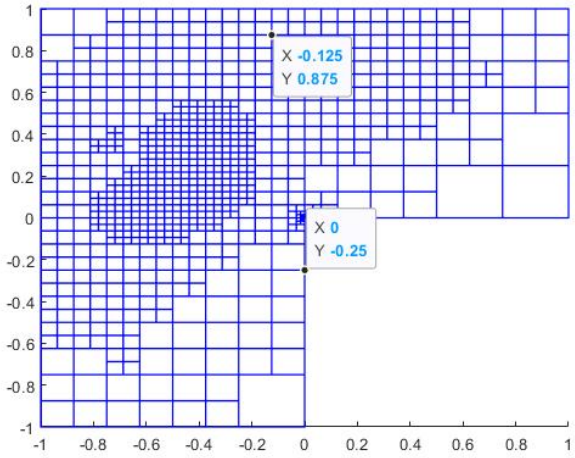

图 33: $\theta = 0.8$,10 次剖分

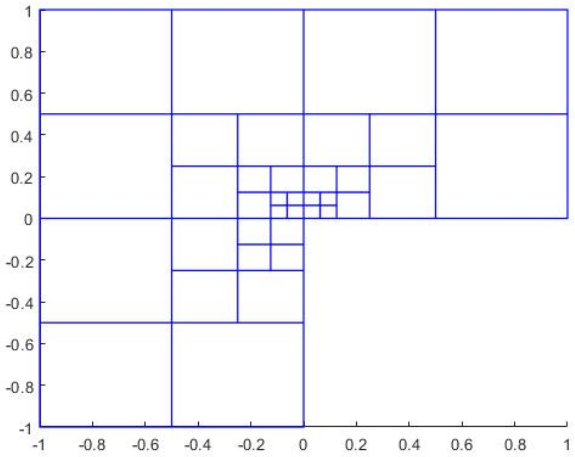

图 34: $\theta = 0.6$,8 次剖分

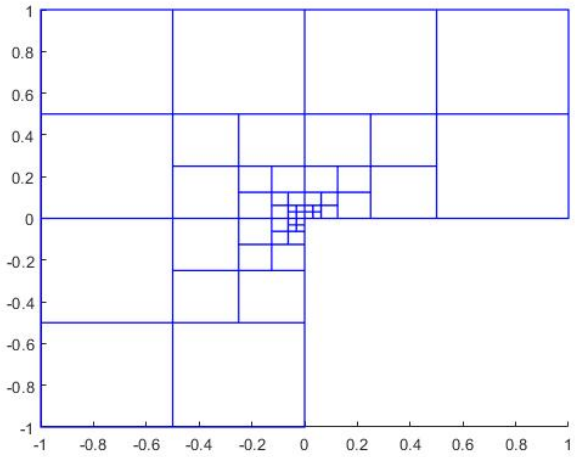

图 35: $\theta = 0.6$,10 次剖分

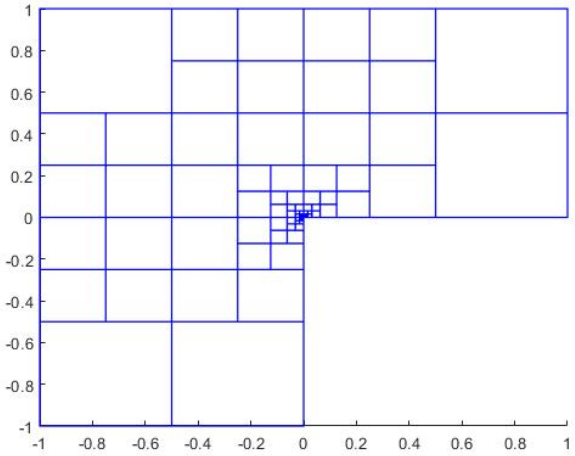

图 36: $\theta = 0.6$,12 次剖分

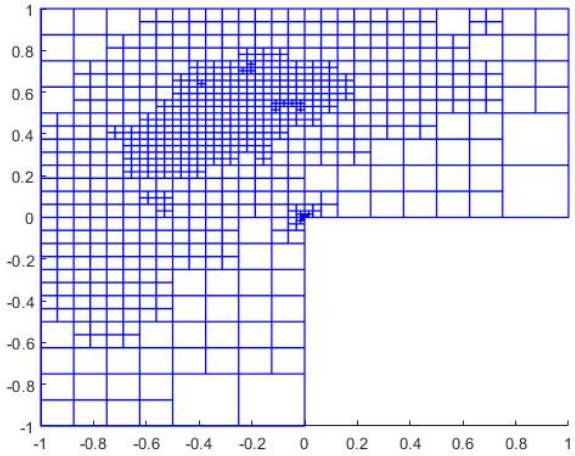

图 37: $\theta = 0.6$,18 次剖分

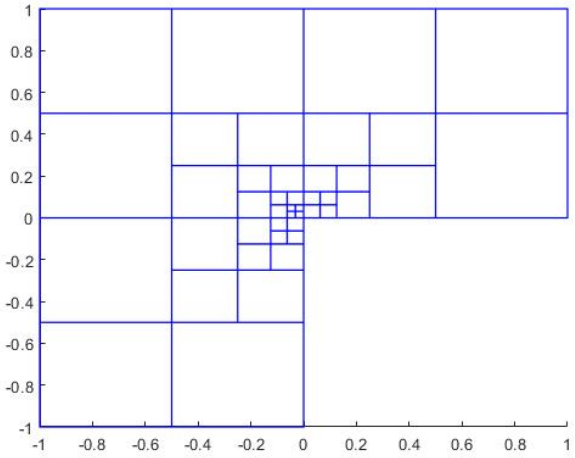

图 38: $\theta = 0.3$,10 次剖分

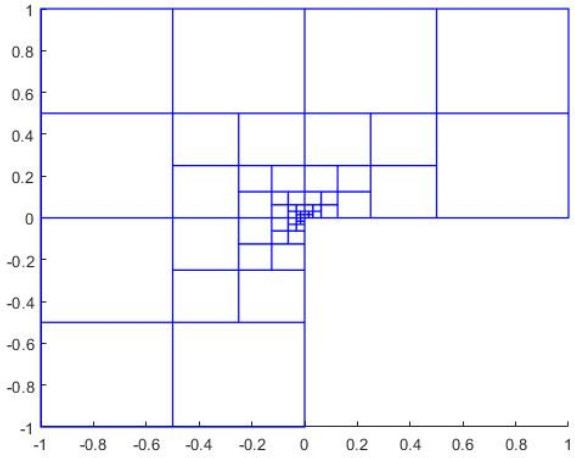

图 39: $\theta = 0.3$,15 次剖分

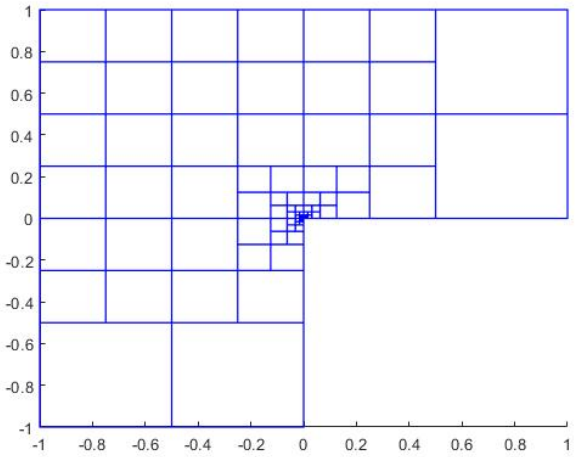

图 40: $\theta = 0.3$,20 次剖分

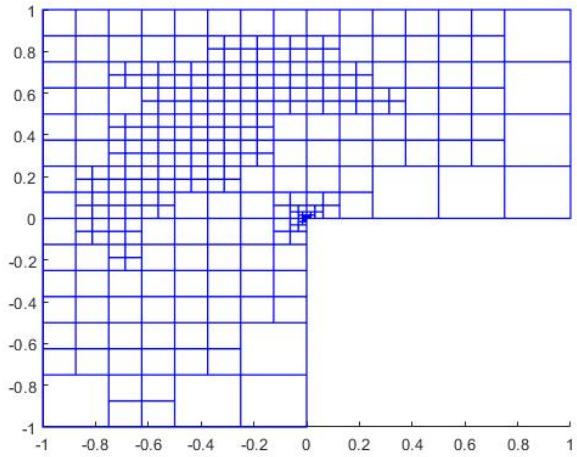

图 41: $\theta = 0.3$,30 次剖分

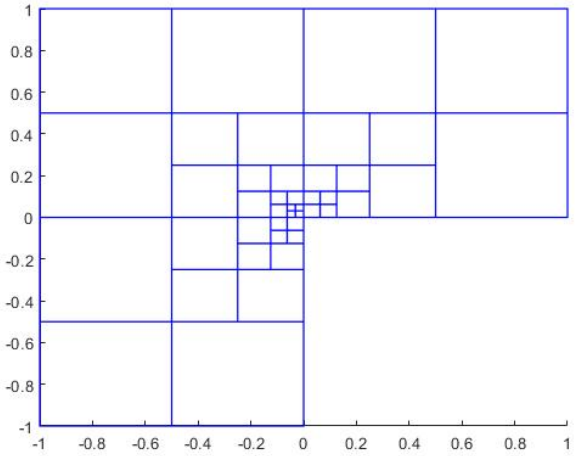

图 42: $\theta = 0.2$,10 次剖分

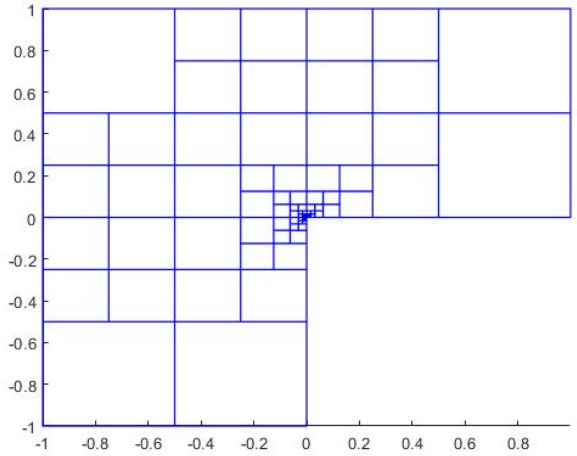

图 43: $\theta = 0.2$,20 次剖分

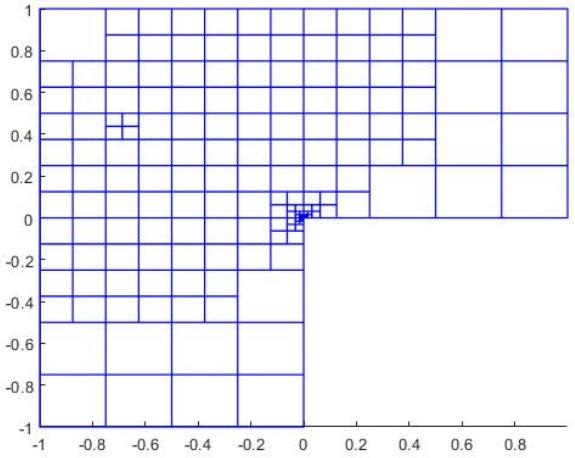

图 44: $\theta = 0.2$,30 次剖分

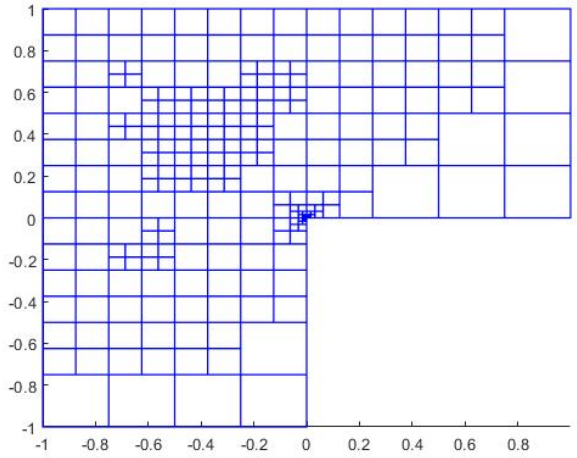

图 45: $\theta = 0.2$,50 次剖分

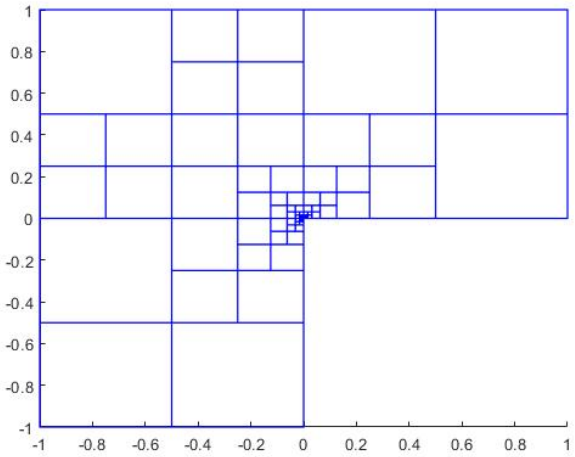

图 46: $\theta = 0.1$,20 次剖分

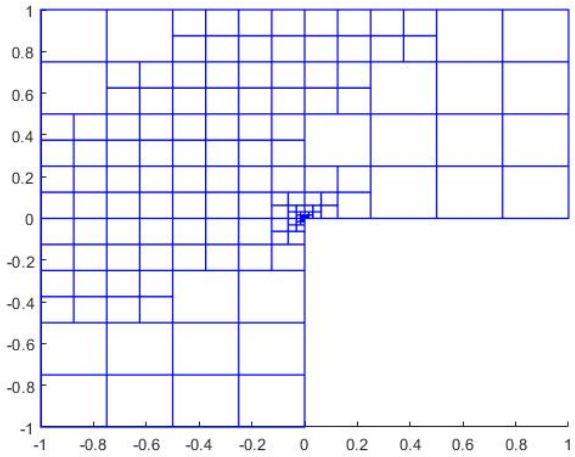

图 47: $\theta = 0.1$,40 次剖分

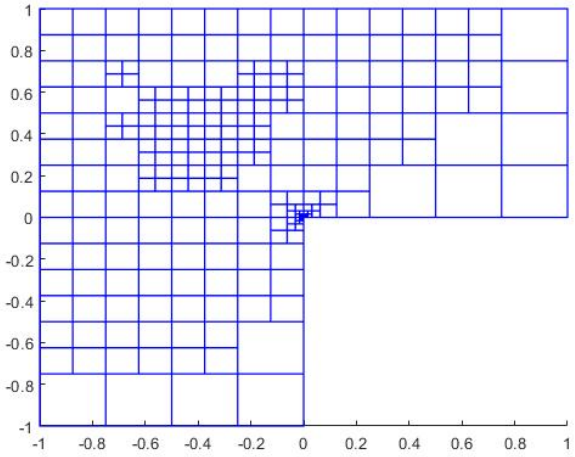

图 48: $\theta = 0.1$,60 次剖分

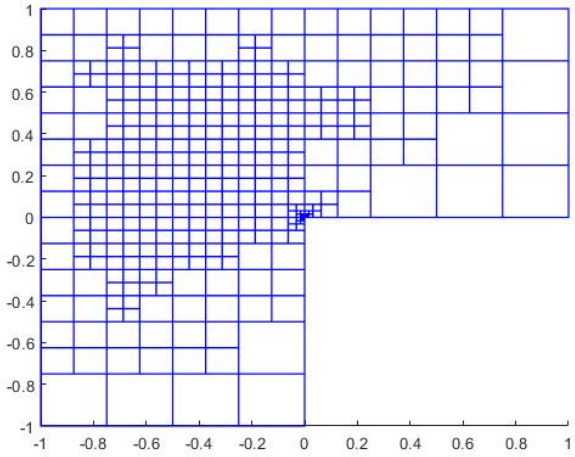

图 49: $\theta = 0.1$,80 次剖分

误差比较

用结点插值的分片线性函数 u_h 逼近原函数，得到误差

$$e_{h,0} = (\int_{\Omega} (u(x,y) - u_h(x,y))^2 dxdy)^{\frac{1}{2}}$$

和

$$e_{h,1} = (\int_{\Omega} |\nabla u(x,y) - \nabla u_h(x,y)|^2 dxdy)^{\frac{1}{2}}$$

对于均匀剖分，计算 $h = 2/N$ 取不同值时的 $\ln e_{h,0}/\ln h$, 对于自适应方法，计算 $\ln e_{h,0}/\ln h_l$, 其中 $h_l = 1/|\Gamma_l|^{\frac{1}{2}}$,$|\Gamma_l|$ 是网格 Γ_l 中的正方形个数。结果如下：

空间步长 $h = 2/N$	$e_{h,0}$	$e_{h,1}$	$\ln e_{h,0}/\ln h$
1/16	3.6513e-3	3.4570	2.0243
1/32	1.3800e-3	1.7254	1.9002
1/64	5.4831e-4	8.5952e-1	1.8054
1/128	2.2694e-4	4.2760e-1	1.7293
1/256	9.7381e-5	2.2001e-1	1.6657

表 4: 均匀剖分不同步长时的误差

单元格数 $\lvert\Gamma_l\rvert$	$e_{h,0}$	$e_{h,1}$	$\ln e_{h,0}/\ln h$
5	0.1237	9.3110e+1	2.5960
16	0.1086	4.6557e+1	1.6014
27	0.1077	2.3284e+1	1.3515
49	5.4296e-2	1.1644e+1	1.4971
131	3.7989e-2	5.8239	1.3416
364	2.1777e-2	2.8210	1.3666
1088	1.4274e-2	1.8207	1.3804

表 5: 自适应网格在不同单元格数时的误差

可以看出，误差与步长、单元格数的平方根大致上是对数线性关系，而且在单元格数相近的情况下，实际上自适应方法的误差比均匀剖分还大。在编写代码的过程中，我所碰到的困难和花费的精力主要在如何判断悬点、进行自适应剖分和建立差分方程上，而解方程则使用的 matlab 基本的求解器。推测可能是在自适应加细的过程中，在原点附近由于梯度趋近于无穷，相对于其他地方就更密，这就导致建立差分方程时，这部分步长的倒数的阶数远大于其他边界部分，同时系数矩阵没有经过很好的整理 (后面的单元格剖分不规则，这使得整理变得十分困难), 解方程的方法是使用的 matlab 基本求解器，可能解方程的舍入误差占了主导地位，而第二题均匀剖分的系数矩阵形如图 25，近似于分块三角阵，形状是很好的，而且各个系数大小相近，使得解得的结果就相对好。因此，自适应方法中再加细很难得到更好的结果，所耗费的时间也远远超过了均匀剖分，除非采用更好的存储系数矩阵和求解的方法。从这次实验看出，简单地套用自适应方法不一定能得到比均匀剖分更好的结果，也应该考虑如何让存储和求解方程更加有效。