差分方法 II

lab1

Wide stencil finite difference method 算例

陈伟

1901110037

2020 年 5 月 20 日

目录

1 问题描述

用宽模版的有限差分法求解 2D 的 Monge-Ampère 方程:

$$\begin{cases} \det D^2 u = f & \text{in } \Omega \subset \mathbb{R}^2 \\ u = g & \text{on } \partial\Omega \end{cases}$$

其中计算区域为单位正方形, 也即是 $\Omega = (0,1)^2$. 令 $\boldsymbol{x} = (x,y)^T, \boldsymbol{x}_0 = (0.5, 05)$. 对于如下三个例子进行数值实验:

- Smooth and radial example:

$$u(\boldsymbol{x}) = \exp(|\boldsymbol{x}|^2/2), \quad f(\boldsymbol{x}) = (1+|\boldsymbol{x}|)^2 \exp(|\boldsymbol{x}|^2).$$

- C^1 example:

$$u(\boldsymbol{x}) = \frac{1}{2}\left((|\boldsymbol{x}-\boldsymbol{x}_0| - 0.2)^+\right)^2, \quad f(\boldsymbol{x}) = \left(1 - \frac{0.2}{|\boldsymbol{x}-\boldsymbol{x}_0|}\right)^+.$$

- Twice differentiable in the interior domain, but has unbounded gradient near the boundary point $(1,1)$:

$$u(\boldsymbol{x}) = -\sqrt{2-|\boldsymbol{x}|^2}, \quad f(\boldsymbol{x}) = 2\left(2-|\boldsymbol{x}|^2\right)^{-2}.$$

边界 $g(\boldsymbol{x})$ 可以通过真解获得. 分别用 explicit solution method 和 Newton's method 来解这个离散的非线性方程. 报告中应包含不同模版的 L^∞ 误差.

2 求解方法

2.1 Wide stencil finite difference method

令

$$\mathrm{MA}[\varphi](x_0) = \min_{(w_1,\cdots,w_d)\in V} \left[\prod_{i=1}^{d}\left(\frac{\partial^2\varphi}{\partial w_i^2}(x_0)\right)^+ - \sum_{i=1}^{d}\left(\frac{\partial^2\varphi}{\partial w_i^2}(x_0)\right)^-\right]$$

在 φ 是凸函数下有:

$$\mathrm{MA}[\varphi] = \det D^2\varphi$$

对于凸区域 Ω, 以及网格剖分, 给定 $\boldsymbol{x}_h \in \Omega$, 以及方向 $\boldsymbol{e}$, 有 $\rho_\pm \in (0,1]$ 使得 $\boldsymbol{x}_h \pm \rho_\pm h\boldsymbol{e} \in \partial\Omega \cup \bar{\Omega}_h$. 当 $\boldsymbol{x}_h \pm h\boldsymbol{e}$ 落在 Ω_h 内部网格点时,$\rho_\pm$ 均为 1; 当落在外边界时,$\rho_\pm$ 为使得 $\boldsymbol{x}_h \pm \rho_\pm h\boldsymbol{e}$ 收缩回边界的

比例. 令

$$\Delta_e u_h(x_h) = \frac{2}{(\rho_+ + \rho_-)|e|^2 h^2}\left[\frac{u_h(x_h+\rho_+ he)-u_h(x_h)}{\rho_+} - \frac{u_h(x_h)-u_h(x_h-\rho_- he)}{\rho_-}\right] \tag{1}$$

以及 $G_\theta \subset (\mathbb{Z}^d)^d$(参数 $d\theta$) 为 $\mathbb{R}^d$ 上的正交基组全体 V 的离散. 并定义:

$$\mathrm{MA}_{h,\theta}^{WS}[u_h](x_h) \triangleq \min_{(v_1,\cdots,v_d)\in G_\theta} \prod_{i=1}^{d} \left(\Delta_{v_i} u_h(x_n)\right)^+$$

对于 Monge-Ampère 方程的宽模版法也即是:

$$\begin{cases} \mathrm{MA}_{h,\theta}^{WS}[u_h](x_h) = f(x_h) & \forall x_h \in \Omega_n \\ u_h(x_h) = g(x_h) & \forall x_h \in \partial\Omega_h \end{cases}$$

对于给定的参数 $\delta > 0$, 将上述 $\mathrm{MA}_{h,\theta}^{WS}$ 换成

$$\mathrm{MA}_{h,\theta,\delta}^{WS}[u_h](x_h) \triangleq \min_{(v_1,\cdots,v_d)\in G_\theta}{}^{\delta} \left[\prod_{i=1}^{d} \left(\Delta_{v_i} u_h(x_h)\right)^{+,\delta}\right]$$

也即得到正则化版本. 总之, 我们要求解方程组 $F[u] = f$, 其中 F 为 $\mathrm{MA}_{h,\theta}^{WS}$ 或者 $\mathrm{MA}_{h,\theta,\delta}^{WS}$, 用以下两种迭代法来求解.

2.2 Explicit solution method

迭代格式为

$$u^{n+1} = u^n + dt(F[u] - f)$$

其中 $dt \sim \mathcal{O}(h^2)$,2D case 的一个选择为:

$$dt = \frac{h^2}{2}\left(\max_{x_h}\max_{(v_1,v_2)\in G_\theta}\left[\left(\Delta_{v_1} u^n(x_h)\right)^+ + \left(\Delta_{v_2} u^n(x_h)\right)^+\right]\right)^{-1}$$

2.3 Newton's method

迭代格式为:

$$u^{n+1} = u^n - \alpha v^n$$

其中 $0 < \alpha < 1$, 为使得残量 $\|F(u^n) - f\|$ 下降的参数. 矫正项 v^n 为满足:

$$\nabla_u F[u^n] v^n = F[u^n] - f$$

3 算法

观察以上方法，不难发现主要分为两部分:

1. $F[u]$ 的建立;
2. 迭代法求解 $F[u] = f$.

下面我们分两部分来构造上述两部分.

3.1 构建 $F[u]$

无论 F 为 $\mathrm{MA}^{WS}_{h,\theta}$ 或者 $\mathrm{MA}^{WS}_{h,\theta,\delta}$, 其中最主要的部分都是:

- G_θ 的构建;
- 对于给定的方向 e, 以及 $x_h \in \Omega_h$, $\Delta_e u_h(x_h)$ 的构建;

然后再根据 $\mathrm{MA}^{WS}_{h,\theta}$ 和 $\mathrm{MA}^{WS}_{h,\theta,\delta}$ 的方式去构建即可.

3.1.1 G_θ 算法

对于给定的模版宽度 WideN(WideN $= 1, 2, 3, \cdots$), 函数 G_θ(WideN) 输出模版宽度为 WideN 的正交基中方向在一象限的方向的集合 $\{\boldsymbol{e} \in \mathbb{Z}^2 : \boldsymbol{e}_1, \boldsymbol{e}_2 = 0, 1, 2, \cdots, N;\ |\boldsymbol{e}| \neq 0\}$, 而且一旦 $\boldsymbol{e}$ 定下来了, 则 $\boldsymbol{e}^\perp = (-\boldsymbol{e}_2, \boldsymbol{e}_1)$ 也就给定了. 下面是 G_θ 的算法.

算法 1 计算 G_θ

输入 WideN
l = 0;
For i = 1:N
  For j = 0:N-1
    l = l+1;
    k = gcd(i,j);
    G_θ(l,1) = i/k, G_θ(l,2) = j/k;
  end
end
删去 G_θ 中以一行为一项来看重复的项
输出 G_θ

3.1.2 $\Delta_e u_h$ 算法

对于给定的方向 e 以及所有内点值 u_h, 我们要计算对应内点的 $\Delta_e u_h$ 的值. 对于 $x_h \in \Omega_h$, 我们先找 $x_{h,\pm} = x_h \pm he$, 若 $x_{h,\pm} \in \Omega_h$, 则 $\rho_\pm = 1$, 若有 $x\pm \notin \Omega_h$, 则根据边界比例来缩小 $\rho_\pm$, 使得

$x_{h,\pm} = x_h \pm \rho_\pm he \in \partial\Omega$. 最后再根据(1), 即可得到 $\Delta_e u_h$. 我们这里给定 u_h 为原来 2D 内点数据按列拉升之后得到的一维数组, 这样我们可以根据 $x_{h,\pm}$ 的是否位于该数组中来判断是否越出边界, 下面给出该算法.(由于后面用 Netwon 法时候可能每个点的方向可能不一样, 故这里我们的 e 可能是 (1,2) 的数组, 也可能是 (Nu,2) 的数组, 其中 Nu 为 u_h 的长度.)

算法 2 计算 $\Delta_e u_h$

输入 u_h, e, h

得到 u_h 的长度 Nu

IF size(e,1)==1

e = repmat(e,size(u_h));

For s=1:Nu

$u_{h,s}$ 对应二维数组的横纵指标 i_s, j_s; 令 $\rho_{s,\pm} = 1$.

计算 $u_{h,s}$ 依赖值 $u_{h,s,\pm}$ 的横纵指标

$$i_{s,\pm} = i_s \pm \rho_{s,\pm} e(s,1), \qquad j_{s,\pm} = j_s \pm \rho_{s,\pm} e(s,2)$$

根据 $i_{s,\pm}, j_{s,\pm}$ 来得到按列拉升后 $u_{h,s,\pm}$ 的位置 $s_\pm$;

如果 $s_\pm$ 均是 $1 : Nu$ 中的某个整数, 则令: $u_{h,\pm,s} = u_{h,s_\pm}$;

否则若 s_+ 超标, 则根据坐标值 $x_{s_+} = (i_{s,+}h, j_{s,+}h)$ 以及 $\partial\Omega$ 来修正收缩比例 ρ_+, 使得重新得到的 $x_{s_+} \in \partial\Omega$, 再令 $u_{h,+,s} = g(x_{s_+})$, 若 s_- 超标, 同理处理.

计算:

$$\Delta_{e_s} u_{h,s} = \frac{2}{(\rho_{+,s}+\rho_{-,s})|e_s|^2 h^2} \left(\frac{u_{h,+,s}-u_{h,s}}{\rho_{+,s}} - \frac{u_{h,s}-u_{h,-,s}}{\rho_{-,s}} \right)$$

end

输出 $\Delta_e u_h$

这样我们得到了模版 G_θ 与 $\Delta_e u_h$ 的算法, 剩下的再根据 $\mathrm{MA}^{WS}_{h,\theta}$ 和 $\mathrm{MA}^{WS}_{h,\theta,\delta}$ 的方式即可构建得到对应的 $F[u]$

3.2 迭代法求解 $F[u] = f$

Explicit solution method 方法按照对应的格式来即可, 我们主要讨论 Newton's method. 其中困难的一步是 $\nabla_u F[u]$ 矩阵的构建. 再进一步，两种方法均会用到的是 $\nabla_u \Delta_e u_h$. 对于不同的点 x_h, 对应的 e 方向可能也不同.

3.2.1 $\nabla_u \Delta_e u_h$ 算法

对于给定的点 x_h, 我们求 $\nabla_u \Delta_e u_h(x_h)$, 而由(1), 我们需要知道对应的 $\rho_\pm$, 以及 $x_h \pm \rho_\pm he$ 的位置. 接算法 2, 我们有:

算法 3 计算 $\nabla_u \Delta_e u_h$

输入 u_h, e, h

得到 u_h 的长度 Nu

IF size(e,1)==1

e = repmat(e,size(u_h));

For s=1:Nu

$u_{h,s}$ 对应二维数组的横纵指标 i_s, j_s; 令 $\rho_{s,\pm} = 1$.

计算 $u_{h,s}$ 依赖值 $u_{h,s,\pm}$ 的横纵指标

$$i_{s,\pm} = i_s \pm \rho_{s,\pm} e(s,1), \qquad j_{s,\pm} = j_s \pm \rho_{s,\pm} e(s,2)$$

根据 $i_{s,\pm}, j_{s,\pm}$ 来得到按列拉升后 $u_{h,s,\pm}$ 的位置 $s_\pm$;

如果 $s_\pm$ 均是 $1:Nu$ 中的某个整数, 则令: $u_{h,\pm,s} = u_{h,s_\pm}$;

否则若 s_+ 超标, 则根据坐标值 $x_{s_+} = (i_{s,+}h, j_{s,+}h)$ 以及 $\partial\Omega$ 来修正收缩比例 ρ_+, 使得重新得到的 $x_{s_+} \in \partial\Omega$, 再令 $u_{h,+,s} = g(x_{s_+})$, 若 s_- 超标, 同理处理.

计算:

$$\nabla_u \Delta_e u_h(s,s) = -\frac{2}{(\rho_{+,s}+\rho_{-,s})|e_s|^2 h^2}\left(\frac{1}{\rho_{+,s}} + \frac{1}{\rho_{-,s}}\right)$$

若 $1 \leqslant s_+ \leqslant Nu$, 则

$$\nabla_u \Delta_e u_h(s,s_+) = -\frac{2}{(\rho_{+,s}+\rho_{-,s})|e_s|^2 h^2}\frac{1}{\rho_{+,s}}$$

若 $1 \leqslant s_- \leqslant Nu$, 则

$$\nabla_u \Delta_e u_h(s,s_-) = -\frac{2}{(\rho_{+,s}+\rho_{-,s})|e_s|^2 h^2}\frac{1}{\rho_{-,s}}$$

end

输出 $\nabla_u \Delta_e u_h$

这样我们就可以分别得到 $\mathrm{MA}_{h,\theta}^{WS}$ 和 $\mathrm{MA}_{h,\theta,\delta}^{WS}$ 的梯度矩阵

3.2.2 $\nabla_u \mathrm{MA}_{h,\theta}^{WS}[u_h]$ 算法

用函数乘积的求导法则我们有:

$$\nabla_u \mathrm{MA}_{h,\theta}^{WS}[u_h] = \sum_{i=1}^{d}\left[\mathbf{diag}\left(\prod_{s\neq i}\Delta_{v_s^*}u_h\right)\nabla_u \Delta_{v_i^*}u_h\right]$$

算法 4 计算 $\nabla_u \mathrm{MA}_{h,\theta}^{WS}[u_h]$

输入 u_h, G_θ, h

令 $\nabla_u \mathrm{MA}_{h,\theta}^{WS}[u_h] = 0$.

在计算 $\nabla_u \mathrm{MA}_{h,\theta}^{WS}[u_h]$ 同时得到每个点的最小正交基 $v^* = (v_1^*, v_2^*, \cdots, v_d^*)$.

For i = 1:d

调用算法 3, 输入 u_h, v_i^*, h, 得到对应的 $\nabla_u \Delta_{v_i^*} u_h$

更新

$$\nabla_u \mathrm{MA}_{h,\theta}^{WS}[u_h] = \nabla_u \mathrm{MA}_{h,\theta}^{WS}[u_h] + \mathbf{diag}\left(\prod_{s\neq i} \Delta_{v_s^*} u_h\right) \nabla_u \Delta_{v_i^*} u_h$$

end

输出 $\nabla_u \mathrm{MA}_{h,\theta}^{WS}[u_h]$

这样我们就得到 $\nabla_u \mathrm{MA}_{h,\theta}^{WS}[u_h]$.

3.2.3 $\nabla_u \mathrm{MA}_{h,\theta,\delta}^{WS}[u_h]$ 算法

由链式法则以及乘积的求导法则以及 $\mathrm{MA}_{h,\theta,\delta}^{WS}[u_h]$ 的表达式, 我们有:

$$\nabla_u \mathrm{MA}_{h,\theta,\delta}^{WS}[u_h] = \partial_x \min^\delta(x,y) \nabla_u x + \partial_y \min^\delta(x,y) \nabla_u y$$

其中

$$x = \prod_{i=1}^{d} (\Delta_{v_i} u_h)^{+,\delta}$$

$$y = \min_{(\hat{v}_1,\cdots,\hat{v}_d)\in G_\theta \backslash (v_1,\cdots,v_d)} \delta \left[\prod_{i=1}^{d} (\Delta_{v_i} u_h)^{+,\delta}\right]$$

不难发现 $\nabla_u y$ 可以递归计算. 所以主要是 $\nabla_u x$ 的计算. 有:

$$\nabla_u x = \sum_{i=1}^{n} \left[\mathbf{diag}\left(\prod_{s=\neq i} (\Delta_{v_s} u_h)^{+,\delta}\right) \nabla_u (\Delta_{v_i} u_h)^{+,\delta}\right]$$

而关于 $\nabla_u (\Delta_{v_i} u_h)^{+,\delta}$, 有

$$\nabla_u (\Delta_{v_i} u_h)^{+,\delta} = \frac{1}{2}\mathbf{diag}\left(1 + \frac{\Delta_{v_i} u_h}{\sqrt{(\Delta_{v_i} u_h)^2 + \delta^2}}\right) \nabla_u (\Delta_{v_i} u_h)$$

这样我们就得到了 $\nabla_u \mathrm{MA}_{h,\theta,\delta}^{WS}[u_h]$ 的算法.

算法 5 计算 $\nabla_u \mathrm{MA}^{WS}_{h,\theta,\delta}[u_h]$

输入 u_h, G_θ, h

得到 G_θ 中向量个数 L

令 $\nabla_u \mathrm{MA}^{WS}_{h,\theta,\delta}[u_h] = 0$, 令 $\nabla_u \mathrm{MA}^{WS}_{h,\theta,\delta}[u_h] = 0$.

For l = 1:L

IF l==1

计算 $\nabla_u \left(\prod_{i=1}^d (\nabla_{v_i^l} u_h)^{+,\delta} \right)$ 并赋值给 $\nabla_u \mathrm{MA}^{WS}_{h,\theta,\delta}[u_h]$

计算 $\prod_{i=1}^d (\nabla_{v_i^l} u_h)^{+,\delta}$, 并赋值给 $\nabla_u \mathrm{MA}^{WS}_{h,\theta,\delta}[u_h]$.

end

计算 $\nabla_u \left(\prod_{i=1}^d (\nabla_{v_i^l} u_h)^{+,\delta} \right)$ 与 $\prod_{i=1}^d (\nabla_{v_i^l} u_h)^{+,\delta}$,

更新

$$\begin{aligned}\nabla_u \mathrm{MA}^{WS}_{h,\theta,\delta}[u_h] = &\frac{1}{2}\mathbf{diag}\left(1 - \frac{\mathrm{MA}^{WS}_{h,\theta,\delta}[u_h] - \prod_{i=1}^d (\nabla_{v_i^l} u_h)^{+,\delta}}{\sqrt{(\mathrm{MA}^{WS}_{h,\theta,\delta}[u_h])^2 + (\prod_{i=1}^d (\nabla_{v_i^l} u_h)^{+,\delta})^2 + \delta^2}}\right) \nabla_u \mathrm{MA}^{WS}_{h,\theta,\delta}[u_h] \\ &+ \frac{1}{2}\mathbf{diag}\left(1 - \frac{\prod_{i=1}^d (\nabla_{v_i^l} u_h)^{+,\delta} - \mathrm{MA}^{WS}_{h,\theta,\delta}[u_h]}{\sqrt{(\mathrm{MA}^{WS}_{h,\theta,\delta}[u_h])^2 + (\prod_{i=1}^d (\nabla_{v_i^l} u_h)^{+,\delta})^2 + \delta^2}}\right) \nabla_u \left[\prod_{i=1}^d (\nabla_{v_i^l} u_h)^{+,\delta}\right]\end{aligned}$$

更新

$$\mathrm{MA}^{WS}_{h,\theta,\delta}[u_h] = \min^\delta \left(\prod_{i=1}^d (\nabla_{v_i^l} u_h)^{+,\delta}, \mathrm{MA}^{WS}_{h,\theta,\delta}[u_h] \right)$$

end

输出 $\nabla_u \mathrm{MA}^{WS}_{h,\theta,\delta}[u_h]$

这样，程序的主要部分就完成了. 后面部分是一些数值算例.

4 数值算例

下面对三个例子分别用 Explicit solution method 和 Newton's method, 选取函数 $u_0(x, y) = x^2 + y^2$ 的值为迭代初值，取迭代终止条件为

$$\|u^{n+1} - u^n\|_\infty \leqslant 1e-8 \text{ or 迭代次数} \geqslant 5000$$

得到的结果如下.

取 $h = 1/8, 1/16, 1/32$, 模版宽度为 $1, 2, 3, 4$.

关于非正则的 Explicit solution method, 对于 Examlpe1, 很遗憾，达迭代上限后也没达收敛精

度. 重新随机选取初值，结果也是如此。总之，需要很大的迭代步此方法才可能收敛.

对于正则的 Explicit solution method，分别取 $\delta = 10, 1, 10^{-2}$, 结果如下

表 1: 正则化 Explicit solution method

example1

	wide\ h:	1/8		1/16		1/32	
		迭代次数	误差	迭代次数	误差	迭代次数	误差
$\delta = 10$	1	1495	1.18e+00	5000	1.18e+00	5000	9.97e-01
	2	665	7.39e-01	3600	7.31e-01	5000	7.05e-01
	3	390	4.26e-01	1965	4.18e-01	5000	4.14e-01
	4	304	2.65e-01	1459	2.57e-01	5000	2.55e-01
	wide\ h:	1/8		1/16		1/32	
		迭代次数	误差	迭代次数	误差	迭代次数	误差
$\delta = 10^{-2}$	1	476	3.55e-03	2253	2.01e-03	5000	1.60e-01
	2	485	1.69e-03	2197	9.94e-04	5000	4.90e-02
	3	497	7.56e-03	2264	1.43e-03	5000	6.41e-02
	4	536	1.52e-02	2316	3.29e-03	5000	7.57e-02

example2

	wide\ h:	1/8		1/16		1/32	
		迭代次数	误差	迭代次数	误差	迭代次数	误差
$\delta = 10$	1	1105	1.38e+00	5000	1.38e+00	5000	1.65e+00
	2	388	7.20e-01	3201	7.14e-01	5000	9.38e-01
	3	171	3.33e-01	1405	3.28e-01	5000	4.60e-01
	4	102	1.47e-01	838	1.42e-01	5000	1.42e-01
	wide\ h:	1/8		1/16		1/32	
		迭代次数	误差	迭代次数	误差	迭代次数	误差
$\delta = 10^{-2}$	1	5000	8.43e-01	5000	1.41e+00	5000	1.72e+00
	2	5000	7.89e-01	5000	1.37e+00	5000	1.71e+00
	3	5000	7.51e-01	5000	1.38e+00	5000	1.71e+00
	4	5000	7.43e-01	5000	1.37e+00	5000	1.71e+00

example3

	wide\ h:	1/8		1/16		1/32	
		迭代次数	误差	迭代次数	误差	迭代次数	误差
$\delta = 10$	1	4196	1.32e+00	5000	2.44e+00	5000	2.27e+00
	2	1724	7.55e-01	5000	1.74e+00	5000	2.18e+00
	3	963	3.97e-01	5000	7.40e-01	5000	2.07e+00
	4	724	2.20e-01	5000	2.56e-01	5000	1.99e+00

可以看出，h 越小，解越不光滑, 则对 $MA^{WS}_{h,\theta,\delta}$ 的正则要求越高，即是需要 δ 越大. 在第三个例子中取 $\delta = 10$ 也不能在 5000 步内收敛, 同时 δ 越大，收敛得到的数值解与真解的误差也就越大. 总的来说效果不理想。下面我们考虑 Netwon 法.

由于非正则化的 Netwon 法得到的梯度矩阵奇异性较大，初值只有落在真解的一个小区间内才收敛。故我们测试正则化的 Netwon 法，同时对应的正则化因子 δ 大时，梯度矩阵奇异性较小，但数值解与真解的误差较大，故我们可以先选取较大的 δ，等用 Netwon 法收敛之后，减小 δ，再用之前得到的数值解作为迭代初值，再用正则化的 Netwon 法迭代至解收敛，再减小 δ(如令 $\delta^{n+1} = \delta^n/2$), 如此重复，直到 δ 充分小或者本次迭代收敛值和初值相差充分小即可。对于三个数值算例, 分别取定 δ_{max}, 取收敛条件为:

$$\delta^n < 1e-6 \text{ or } \|u^n - u^{n-1}\|_\infty < 1e-8$$

其中 u^n 为以 u^{n-1} 为初值,δ 取为 δ^n 的收敛结果.

表 2: 正则化 Explicit solution method

example1

	wide\ h:	1/16		1/32		1/64	
		迭代次数	误差	迭代次数	误差	迭代次数	误差
$\delta_{max} = 2$	1	171	2.03e-03	177	1.69e-03	180	1.61e-03
	2	179	1.07e-03	186	6.37e-04	190	4.92e-04
	3	184	1.42e-03	192	4.27e-04	202	2.51e-04
	4	183	3.27e-03	199	6.02e-04	217	2.09e-04

example2

	wide\ h:	1/8		1/16		1/32	
		迭代次数	误差	迭代次数	误差	迭代次数	误差
$\delta_{max} = 5$	1	1092	2.59e-03	558	2.38e-03	1150	2.13e-03
	2	487	1.71e-03	384	8.02e-04	466	6.63e-04
	3	370	1.15e-03	334	5.65e-04	348	3.12e-04
	4	340	1.84e-03	338	4.53e-04	368	2.42e-04

example3

	wide\ h:	1/8		1/16		1/32	
		迭代次数	误差	迭代次数	误差	迭代次数	误差
$\delta = 5$	1	315	4.45e-03	486	1.58e-03	10	4.67e+06
	2	240	4.40e-03	273	1.58e-03	25	1.79e+07
	3	225	1.97e-03	258	1.03e-03	20	4.80e+05
	4	226	3.99e-03	257	1.46e-03	42	2.28e+06
	wide\ h:	1/8		1/16		1/32	
		迭代次数	误差	迭代次数	误差	迭代次数	误差
$\delta = 10$	1	492	4.45e-03	778	1.58e-03	1561	8.62e-04
	2	314	4.40e-03	416	1.58e-03	578	5.59e-04
	3	271	1.97e-03	313	1.03e-03	351	5.59e-04
	4	261	3.99e-03	302	1.46e-03	360	3.45e-04

可以看出,Netwon 法的收敛速度比显示法快很多。但是 Netwon 法得到的矩阵可能有较强的奇

异性，只有在初值选取较好时才能较好收敛，为此好的初值选取很重要。

5 代码说明

本次程序使用 matlab 编写，包含 main.m 和 Readme.txt 等在内一共有 24 个文件，在文件夹‘matlabcode’中. 关于各文件的含义以及 main.m 的运行参见 Readme.txt.