Implement of Two-scale Scheme for the Monge-Ampère Equation

Bowen Shi

June, 2023

Codes include functions and 4 .mlx files which implement the monotone operator(two_scale_newton.mlx and tow_scale_perron.mlx), accurate operator(two_scale_accurate.mlx) and filter scheme(two_scale_filter.mlx).

1 Introduction to Two-Scale Scheme

We consider the Monge-Ampère equation with Dirichlet boundary condition:

$$\begin{cases} \det D^2 u = f & \text{in } \Omega \subset \mathbb{R}^d, \\ u = g & \text{on } \partial\Omega \end{cases} \tag{1}$$

where Ω is a uniformly convex domain and $f \geq 0$ and g are uniformly continuous functions. We seek a convex solution u, which is critical for the upper equation to be elliptic and have a unique viscosity solution.

If we rewrite the determinant of D^2u into

$$\det D^2 w(x) = \min_{\boldsymbol{v} \in \mathbb{S}^\perp} \prod_{j=1}^{d} v_j^T D^2 w(x) v_j$$

where $\mathbb{S}^\perp$ is the set of all d-orthonormal bases $\boldsymbol{v} = (v_j)_{j=1}^d, v_j \in \mathbb{R}^d$. The minimum here is achieved by the eigenvectors of $D^2w(x)$ and is equal to the product of the respective eigenvalues. We can discretize the above formula in various ways, employing different polynomial spaces and approximations for the directional derivatives given by $v_j^T D^2 w v_j$. These choices lead to schemes with different theoretical properties and levels of accuracy. We now give a brief introduction the discretization used in [1][2].

First, we introduce our settings and notations. We discretize the domain Ω by a shape regular and quasi-uniform mesh $\mathcal{T}_h^1$ with spacing h, the fine scale, and construct a space $\mathbb{V}_h^1$ of continuous piecewise linear functions over $\mathcal{T}_h^1$. The superscript 1 of $\mathbb{V}_h^1$ indicates the use of linear polynomials whereas that of $\mathcal{T}_h^1$ entails the use of straight (affine equivalent) simplices. We denote by Ω_h the computational domain, namely the union of the elements. We also denote by $\mathcal{N}_h$ the nodes of $\mathcal{T}_h$, and by

$$\mathcal{N}_h^b := \{x_i \in \mathcal{N}_h : x_i \in \partial\Omega_h\}, \quad \mathcal{N}_h^0 := \mathcal{N}_h \backslash \mathcal{N}_h^b$$

the boundary and interior nodes, respectively. We require that $\mathcal{N}_h^b \subset \partial\Omega$, which in view of the convexity of Ω implies that Ω_h is also convex and $\Omega_h \subset \Omega$.

1.1 Monotone Operator

We define the second and coarser δ_m is the length of directions we use to approximate second directional derivatives by central second order differences:

$$\nabla^2_{\delta_m} w(x; v) := \frac{w(x + \delta_m v) - 2w(x) + w(x - \delta_m v)}{|v|^2 \delta_m^2} \quad \text{and} \quad |v| \leq 1$$

for any $w \in C^0(\bar{\Omega})$. Let $\varepsilon = (h, \delta_m, \theta_m)$ represent the two scales and a third parameter θ_m that is utilized to discretize $\mathbb{S}^\perp$ with precision θ_m. We ask that for any v in the unit sphere $\mathbb{S}$, there exists v^{θ_m} that belongs in our discrete approximate set $\mathbb{S}_{\theta_m}$ such that

$$\left|v - v^{\theta_m}\right| \leq \theta_m$$

Likewise, we define the finite set $\mathbb{S}^\perp_{\theta_m}$: for any $\mathbf{v}^{\theta_m} = \left(v_j^{\theta_m}\right)_{j=1}^d \in \mathbb{S}^\perp_{\theta_m}, v_j^{\theta_m} \in \mathbb{S}_{\theta_m}$ and there exists $\mathbf{v} = (v_j)_{j=1}^d \in \mathbb{S}^\perp$ such that $\left|v_j - v_j^{\theta_m}\right| \leq \theta_m$ for all $1 \leq j \leq d$ and conversely. We can now define the discrete

monotone operator to be

$$T_{\varepsilon,m}[w](x_i) := \min_{\mathbf{v}\in\mathbb{S}^{\perp}_{\theta_m}} \left(\prod_{j=1}^{d} \nabla^{2,+}_{\delta_m} w(x_i; v_j) - \sum_{j=1}^{d} \nabla^{2,-}_{\delta_m} w(x_i; v_j) \right) \tag{2}$$

where $\nabla^{2,+}_{\delta_m}$ and $\nabla^{2,-}_{\delta_m}$ denote the positive and negative parts of $\nabla^2_{\delta_m}$ respectively and $x_i \in \mathcal{N}^0_h$. The discrete solution $u_\varepsilon \in \mathbb{V}^1_h$ satisfies

$$T_{\varepsilon,m}[u_\varepsilon](x_i) = f(x_i) \quad \forall x_i \in \mathcal{N}^0_h, \quad u_\varepsilon(x_i) = g(x_i) \quad \forall x_i \in \mathcal{N}^b_h.$$

1.2 Accurate Operator

In order to achieve a better interpolation error and a more accurate discretization of second directional derivatives, we utilize quadratic polynomial interpolation. To this end, we introduce again two scales h and δ_a, where $\delta_a \neq \delta_m$ is the coarse scale corresponding to the length of directions used for accurate discretization of second derivatives. We define the space $\mathbb{V}^2_h$ of continuous, piecewise quadratic functions. For a more general domain, we could use also use isoparametric finite elements to get a better approximation of the boundary. We also employ a more accurate approximation of the second directional derivatives that relies on five, rather than three, point stencils. Consequently, second differences for $u_\varepsilon \in \mathbb{V}^2_h$ are now given by

$$\nabla^2_{\delta_a} u_\varepsilon(x_i; v) := \frac{-u_\varepsilon(x_i + \delta_a v) + 16u_\varepsilon\left(x_i + \frac{\delta_a}{2}v\right) - 30u_\varepsilon(x_i) + 16u_\varepsilon\left(x_i - \frac{\delta_a}{2}v\right) - u_\varepsilon(x_i - \delta_a v)}{3\delta_a^2|v|^2}$$

where $x_i \in \mathcal{N}^0_h$ and $v \in \mathbb{S}_{\theta_a}$. The symbol $\mathbb{S}_{\theta_a}$ indicates that we use a different angle discretization parameter θ_a for the accurate operator. The accurate scheme then becomes: We seek $u_\varepsilon \in \mathbb{V}^2_h$ such that $u_\varepsilon(x_i) = g(x_i)$ for $x_i \in \mathcal{N}^b_h$ and for $x_i \in \mathcal{N}^0_h$

$$T_{\varepsilon,a}[u_\varepsilon](x_i) := \min_{\mathbf{v}\in\mathbb{S}^{\perp}_{\theta_a}} \left(\prod_{j=1}^{d} \nabla^{2,+}_{\delta_a} u_\varepsilon(x_i; v_j) - \sum_{j=1}^{d} \nabla^{2,-}_{\delta_a} u_\varepsilon(x_i; v_j) \right) = f(x_i) \tag{3}$$

We obbserve that this scheme is no longer monotone. As a result, a method relying only on this discretization cannot be proven to converge to viscosity solutions.

1.3 Filter Scheme

The idea is to use of a filter function that combines the accurate and the monotone operator and guarantees that the monotone operator will be used if the accurate operator fails, due to the lack of monotonicity.

We start with the two meshes and function spaces used. Let $\mathcal{T}^1_h$ be a shape regular and quasi-uniform mesh of size h and $\mathbb{V}^1_h$ be the corresponding space of continuous piecewise linear elements. Let $\mathcal{T}^2_{2h}$ be a mesh of size $2h$ with same nodes as $\mathcal{T}^1_h$ and $\mathbb{V}^2_{2h}$ be the corresponding space of continuous piecewise quadratic elements; see Figure 1 and Figure 2.

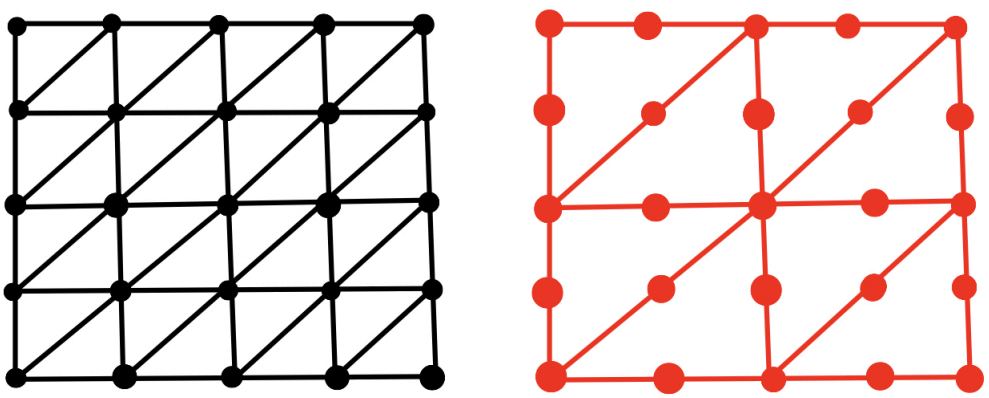

Figure 1: mesh $\mathcal{T}^1_h$(left) and $\mathcal{T}^2_{2h}$(right) share same nodes

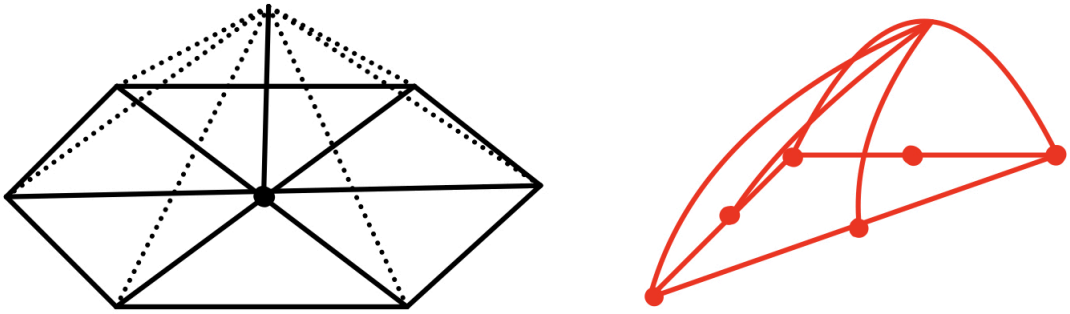

Figure 2: basis function of $\mathbb{V}_h^1$(left) and $\mathbb{V}_{2h}^2$(right)

An important consequence of this two-grid approach is that functions in $\mathbb{V}_h^1$ and $\mathbb{V}_h^2$ have degrees of freedom at the same nodes, including mid-points on the boundary of Ω_{2h}^2.

We now exploit this structure as follows. Let $\mathbf{U}_\varepsilon = \left\{U_\varepsilon^i\right\}_i \in \mathbb{R}^N$ be a grid function where N is the number of nodes of either $\mathbb{V}_h^1$ or $\mathbb{V}_{2h}^2$. We define two functions $u_\varepsilon^1 \in \mathbb{V}_h^1$ and $u_\varepsilon^2 \in \mathbb{V}_{2h}^2$ with nodal values dictated by $\mathbf{U}_\varepsilon$

$$u_\varepsilon^1(x_i) = u_\varepsilon^2(x_i) = U_\varepsilon^i \quad \forall x_i \in \mathcal{N}_h$$

and compare them via a filter function F; see Section 2 for an explicit definition. We thus seek $\mathbf{U}_\varepsilon \in \mathbb{R}^N$ such that $U_\varepsilon^i = g(x_i)$ for all $x_i \in \mathcal{N}_h^b$ and for all $x_i \in \mathcal{N}_h^0$

$$T_{\varepsilon,f}[\mathbf{U}_\varepsilon](x_i) := T_{\varepsilon,m}\left[u_\varepsilon^1\right](x_i) + \tau F\left(\frac{T_{\varepsilon,a}\left[u_\varepsilon^2\right](x_i) - T_{\varepsilon,m}\left[u_\varepsilon^1\right](x_i)}{\tau}\right) = f(x_i) \tag{4}$$

The filter function F is required to be

- compactly supported, so monotone operator dominates when error between two operators is too large
- continuous, hence values are uniformly bounded
- equal to the identity close to the origin, so accurate operator dominates when error between two operators is small
- $\tau(\varepsilon)$ depends on the scales $\varepsilon = (h, \delta_a, \delta_m, \theta_a, \theta_m)$ of the accurate and monotone operators

Here we define two filter function, $F_\sigma(s)$ and $\tilde{F}_\sigma(s)$

$$F_\sigma(s) := \begin{cases} s, & |s| \le 1 \\ 0, & |s| \ge 1+\sigma \\ -\frac{1}{\sigma}s + \frac{1+\sigma}{\sigma}, & 1 < s < 1+\sigma \\ -\frac{1}{\sigma}s - \frac{1+\sigma}{\sigma}, & -1-\sigma < s < -1. \end{cases}$$

$$\widetilde{F}_\sigma(s) := \begin{cases} s, & -1 \le s \le 0 \\ -\frac{1}{\sigma}s - \frac{1+\sigma}{\sigma}, & -1-\sigma \le s \le -1 \\ 0, & \text{otherwise.} \end{cases}$$

We choose $F_\sigma(s)$ as filter when $f(x_i) > 0$ and $\tilde{F}_\sigma(s)$ when $f(x_i) = 0$. This decision is to make sure u_ϵ^1 is always discrete convex, and at the same time the asymmetric property of $\tilde{F}_\sigma(s)$ doesn't affect the solution too much. For detailed explanation, see[2].

2 Nonlinear Solvers

2.1 Semi-smooth Newton Method

We solve the nonlinear algebraic equation (2),(4) via a damped semi-smooth Newton iteration. Let $\mathbf{z} := (z_h(x_i))_{i=1}^N \in \mathbb{R}^N$ stand for the vector of nodal values of a generic $z_h \in \mathbb{V}_h$; thus N is the cardinality of

$\mathcal{N}_h^0$. If $\mathbf{u}_n = (u_\varepsilon^n(x_i))_{i=1}^N, \mathbf{DT}_\varepsilon[\mathbf{u}_n]$ is the Jacobian matrix of the nonlinear map $\mathbf{T}_\varepsilon : \mathbb{R}^N \to \mathbb{R}^N$ at $\mathbf{u}_n$, and $\mathbf{f} = (f(x_i))_{i=1}^N$, then a Newton increment is given by

$$\mathbf{DT}_\varepsilon[\mathbf{u}_n]\mathbf{w}_n = \mathbf{f} - \mathbf{T}_\varepsilon[\mathbf{u}_n]$$

and the n-th Newton step by

$$\mathbf{u}_{n+1} = \mathbf{u}_n + \tau \mathbf{w}_n,$$

where the damping parameter $\tau \in (0,1]$, which might depend on n, satisfies

$$\|f - T_\varepsilon[u_\varepsilon^n + \tau w_n]\|_{L^2(\Omega)} < \|f - T_\varepsilon[u_\varepsilon^n]\|_{L^2(\Omega)}.$$

We now explain the construction of $\mathbf{DT}_\varepsilon[\mathbf{u}_n]$.

2.1.1 Monotone Operator

The operator $\mathbf{T}_\varepsilon$ reads

$$\mathbf{T}_\varepsilon[\mathbf{z}](i) := \min_{\mathbf{v}=(v_j)_{j=1}^d \in \mathbb{S}_\theta^\perp} \left(\prod_{j=1}^d \nabla_\delta^{2,+}\mathbf{z}(i;v_j) - \sum_{j=1}^d \nabla_\delta^{2,-}\mathbf{z}(i;v_j) \right) = f(x_i)$$

according to (2). Let now $\mathbf{v}^i = \left(v_j^i\right)_{j=1}^d \in \mathbb{S}_\theta^\perp$ be a set of directions that realize the minimum of $\mathbf{T}_\varepsilon[\mathbf{u}_n](i)$ and denote $\mathbf{V} := \left(\mathbf{v}^i\right)_{i=1}^N \in \mathbb{R}^{d\times d\times N}$, the collection of the minimizing d-tuples $\mathbf{v}^i$ for all $i = 1,\dots N$. Combining the above notation, we denote the matrix that contains the j-th minimizing directions for each node by $\mathbf{V}_j \in \mathbb{R}^{d\times N}$. So now we have:

$$\mathbf{T}_\varepsilon[\mathbf{u}_n] = \bigodot_{j=1}^d \nabla_\delta^{2,+}\mathbf{u}_n(\mathbf{V}_j) - \sum_{j=1}^d \nabla_\delta^{2,-}\mathbf{u}_n(\mathbf{V}_j)$$

where $\odot$ stands for the component-wise multiplication of vectors. Using left derivative of the max and min function, we can then obtain

$$\mathbf{D}\left[\nabla_\delta^{2,+}\mathbf{u}_n(\mathbf{V}_j)\right]\mathbf{w}_n = \mathbf{H}^+\left[\nabla_\delta^2\mathbf{u}_n(\mathbf{V}_j)\right] \odot \nabla_\delta^2\mathbf{w}_n(\mathbf{V}_j)$$

and Let now $\mathbf{v}^i = \left(v_j^i\right)_{j=1}^d \in \mathbb{S}_\theta^\perp$ be a set of directions that realize the minimum of $\mathbf{T}_\varepsilon[\mathbf{u}_n](i)$ and denote $\mathbf{V} := \left(\mathbf{v}^i\right)_{i=1}^N \in \mathbb{R}^{d\times d\times N}$, the collection of the minimizing d-tuples $\mathbf{v}^i$ for all $i = 1,\dots N$. Combining the above notation, we denote the matrix that contains the j-th minimizing directions for each node by $\mathbf{V}_j \in \mathbb{R}^{d\times N}$. This allows us to display our Jacobian in a vectorized form, using the notation $\nabla_\delta^2\mathbf{u}_n(\mathbf{V}_j) := \left(\nabla_\delta^2\mathbf{u}_n\left(i;v_j^i\right)\right)_{i=1}^N$, since (6.1) gives for $\mathbf{z} = \mathbf{u}_n$:

$$\mathbf{T}_\varepsilon[\mathbf{u}_n] = \bigodot_{j=1}^d \nabla_\delta^{2,+}\mathbf{u}_n(\mathbf{V}_j) - \sum_{j=1}^d \nabla_\delta^{2,-}\mathbf{u}_n(\mathbf{V}_j)$$

where $\odot$ stands for the component-wise multiplication of vectors. Using left derivative of the max and min function, we can then obtain

$$\mathbf{D}\left[\nabla_\delta^{2,+}\mathbf{u}_n(\mathbf{V}_j)\right]\mathbf{w}_n = \mathbf{H}^+\left[\nabla_\delta^2\mathbf{u}_n(\mathbf{V}_j)\right] \odot \nabla_\delta^2\mathbf{w}_n(\mathbf{V}_j)$$

and

$$\mathbf{D}\left[\nabla_\delta^{2,-}\mathbf{u}_n(\mathbf{V}_j)\right]\mathbf{w}_n = \mathbf{H}^-\left[\nabla_\delta^2\mathbf{u}_n(\mathbf{V}_j)\right] \odot \nabla_\delta^2\mathbf{w}_n(\mathbf{V}_j)$$

Here $\mathbf{H}^+$is the operator that assigns 1 to a strictly positive component and zero otherwise, $\mathbf{H}^-$assigns -1 to a nonpositive component and 0 otherwise. Now we have

$$\mathbf{DT}_\varepsilon[\mathbf{u}_n]\mathbf{w}_n = \sum_{j=1}^d \boldsymbol{\nabla}_\delta^2\mathbf{w}_n(\mathbf{V}_j) \bigodot \left(\mathbf{H}^+\left[\boldsymbol{\nabla}_\delta^2\mathbf{u}_n(\mathbf{V}_j)\right] \bigodot_{k\neq j} \boldsymbol{\nabla}_\delta^{2,+}\mathbf{u}_n(\mathbf{V}_k) - \mathbf{H}^-\left[\boldsymbol{\nabla}_\delta^2\mathbf{u}_n(\mathbf{V}_j)\right] \right),$$

2.1.2 Filter Scheme

Note that we could compute the Jacobian matrix of the accurate operator using the same procedure as the monotone operator. From (4), again using left derivative of the fuction F, we get

$$\mathbf{DT}_{\varepsilon,f}[\mathbf{U}_\varepsilon] := \mathbf{DT}_{\varepsilon,m}\left[\mathbf{u}_\varepsilon^1\right] + F'\left(\frac{\mathbf{T}_{\varepsilon,a}\left[\mathbf{u}_\varepsilon^2\right] - \mathbf{T}_{\varepsilon,m}\left[\mathbf{u}_\varepsilon^1\right]}{\tau}\right)\left(\mathbf{DT}_{\varepsilon,m}\left[\mathbf{u}_\varepsilon^2\right] - \mathbf{DT}_{\varepsilon,m}\left[\mathbf{u}_\varepsilon^1\right]\right).$$

2.2 Perron Iteration for Monotone Operators

First, we choose

$$q(x) = \frac{1}{2}\|f\|_{L^\infty(\Omega)}^{1/d}|x|^2$$

to obtain that for $q_h = \mathcal{I}_h q$ and for all $x_i \in \mathcal{N}_h^0$

$$T_\varepsilon\left[q_h\right](x_i) \geq \|f\|_{L^\infty(\Omega)} \geq f(x_i)$$

Let w be a convex function with Dirichlet condition $w = g - q$, whence $w_h := \mathcal{I}_h w$ satisfies

$$T_\varepsilon\left[w_h\right](x_i) \geq 0 \quad \forall x_i \in \mathcal{N}_h^0$$

and $w_h = \mathcal{I}_h g - q_h$ on $\mathcal{N}_h^b$. We define the initial iterate to be

$$u_h^0 := w_h + q_h$$

and note that u_h^0 is discretely convex and satisfies the Dirichlet condition $u_h^0 = \mathcal{I}_h g$ on $\partial\Omega_h$. Since all the terms in $T_\varepsilon\left[u_h^0\right](x_i)$ are nonnegative, we also deduce

$$T_\varepsilon\left[u_h^0\right](x_i) = \min_{\mathbf{v}\in\mathbb{S}_\theta^\perp} \prod_{j=1}^{d} \left(\nabla_\delta^2 w_h(x_i; v_j) + \nabla_\delta^2 q_h(x_i; v_j)\right) \geq f(x_i) \quad \forall x_i \in \mathcal{N}_h^0.$$

Next, we use Perron loop to lift node values one by one to make

$$T_\varepsilon\left[u_h^{k,i}\right](x_i) = f(x_i) \quad x_i \text{ traversal } \mathcal{N}_h^0.$$

and set

$$u_h^{k+1} = u_h^{k,N}.$$

Due to the monotonicity of the operator, we know this could be solved by classical numerical algorithm for nonlinear equations(like the bisection algorithm).

3 Implement Techniques

3.1 Initial Value

No matter which algorithm we use to solve the nonlinear system, a propriate initial value is nessassary. We initialize the Newton iteration with $\mathbf{u}_0$ corresponding to the Galerkin solution in $\mathbb{V}_h$ to the auxiliary problem $\Delta u_0 = (d!f)^{1/d}$ in Ω and $u_0 = g$ on $\partial\Omega$, as proposed in [1], but only for the coarser mesh $h = 2^{-4}$.(For some cases with sigularity, interpolation might lead to unconvergence, so we still need to calculate directly on fine mesh). For all subsequent refinements we interpolate the discrete solution in the previous coarse mesh and use it as initial guess. This greatly improves the residual error and leads to minimal or no damping. When we're considering quadratic functions, we use quadratic interpolation instead.

Note that in Perron iteration, we need to find a convex(not just discrete convex) function satisfying certain boundary conditions. Certainlly, we could adopt the upper algorithm to solve a Laplace equation. However, this calculation does not guarantee convexity, thus convexifying initial value is a must. To achieve this, we use MATLAB function convhulln to calculate the convex hull of the 3D solution, and then remove the top parts and the side parts. This leads to the a convex envelope, and we use this as ω_h in section 2.2. Numerical tests show that convexifying the interpolation solution on a finer mesh also reduces Newton iteration number.

3.2 Jacobian Assembling

To calculate the Jacobian of the operator $\mathbf{T}$, we must calculate the coefficients of node values concerning central differences. Luckily, the use of a standard grid makes it easy to locate $x_i + \delta v$ and $x_i - \delta v$(if it's out of the domain, we need to do a maximum truncation). After knowing which unit it falls in, we use barycentric coordinates to calculate the coefficients directly. The matrix is stored by sparse structure and solved by MATLAB back slash operator.

4 Numerical Experiments

For **Monotone Operator, Accurate Operator and Perron Iteration**, we stop the Newton iterations when

$$\left\|f - T_\varepsilon\left[u_\varepsilon^{n+1}\right]\right\|_{L^2(\Omega)} < 10^{-8}\left\|f - T_\varepsilon\left[u_\varepsilon^{0}\right]\right\|_{L^2(\Omega)}.$$

For **Filter Scheme**, we observe that it only takes a few steps to reach very high accuracy, so we stop iterations when

$$\|f - T_\varepsilon\left[u_\varepsilon^{n}\right]\|_{L^2(\Omega)} < 0.3 \quad \text{or iteration numbers} > 8$$

4.1 Smooth Hessian

We choose the solution u and force f to be

$$u(x) = e^{|x|^2/2}, \quad f(x) = \left(1 + |x|^2\right)e^{|x|^2} \quad \forall x \in \Omega.$$

We choose $\delta, \theta \approx h^{1/2}$. Threshhold in filter scheme is chosen as $4e^2h$.

Perron Iteration:

Mesh size	number of points on circle	$L_\infty-$ error	Perron steps
$h = 2^{-4}$	20	$1.1\,10^{-2}$	33
$h = 2^{-5}$	28	$5.7\,10^{-3}$	57

Monotone Operator:

Mesh size	number of points on circle	$L_\infty-$ error	Newton steps
$h = 2^{-4}$	20	$5.4\,10^{-4}$	19
$h = 2^{-5}$	28	$3.0\,10^{-4}$	7
$h = 2^{-6}$	44	$1.4\,10^{-4}$	8

Accurate Operator:

Mesh size	number of points on circle	$L_\infty-$ error	Newton steps
$h = 2^{-4}$	20	$3.6\,10^{-4}$	14
$h = 2^{-5}$	28	$1.4\,10^{-4}$	6
$h = 2^{-6}$	44	$1.1\,10^{-4}$	7

Filter Scheme:

Mesh size	number of points on circle	$L_\infty-$ error	Newton steps	Active Sets
$h = 2^{-4}$	20	$2.3\,10^{-3}$	13	15
$h = 2^{-5}$	28	$7.4\,10^{-4}$	9	18
$h = 2^{-6}$	44	$1.1\,10^{-4}$	2	20
$h = 2^{-7}$	64	$3.8\,10^{-5}$	2	149

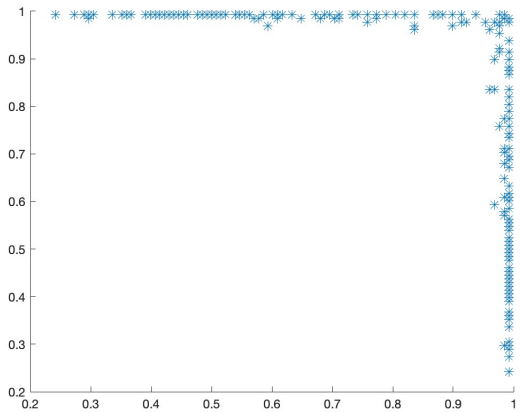

Figure 3: Active sets of the last iteration in case 2^7

The active set for $\tau = 6e^2h$ and $h = 2^{-7}$ is displayed in Figure 3. We observe that active sets are gathered close to the upper-right corner, where u and its derivatives are larger.

4.2 Discontinous Hessian

We choose the solution u and forcing function f to be

$$u(x) = \frac{1}{2}\left(\max\left(|x - x_0| - 0.2, 0\right)\right)^2, \quad f(x) = \max\left(1 - \frac{0.2}{|x - x_0|}, 0\right) \quad \forall x \in \Omega$$

where $x_0 = (0.5, 0.5)$. Since $f = 0$ in the ball centered at x_0 of radius 0.2 , this example is degenerate elliptic. we choose $\delta \approx h^{4/5}, \theta \approx h^{2/5}$. Threshhold in filter scheme is chosen as $0.41h^{2/5}$.

Perron Iteration:

Mesh size	number of points on circle	$L_\infty-$ error	Perron steps
$h = 2^{-4}$	12	$1.1\,10^{-2}$	35
$h = 2^{-5}$	20	$6.5\,10^{-3}$	59

Monotone Operator:

Mesh size	number of points on circle	$L_\infty-$ error	Newton steps	Damping
$h = 2^{-4}$	12	$7.3\,10^{-3}$	13	0.8
$h = 2^{-5}$	20	$4.0\,10^{-3}$	14	0.8
$h = 2^{-6}$	28	$2.8\,10^{-3}$	14	0.8

Accurate Operator:

Mesh size	number of points on circle	$L_\infty-$ error	Newton steps	Damping
$h = 2^{-4}$	12	$1.3\,10^{-3}$	13	0.8
$h = 2^{-5}$	20	$3.7\,10^{-4}$	16	0.8
$h = 2^{-6}$	28	$1.2\,10^{-4}$	16	0.8

Filter Scheme:

Mesh size	number of points on circle	$L_\infty-$ error	Newton steps	Damping	Active Sets
$h = 2^{-4}$	12	$1.4\,10^{-3}$	3	0.8	0
$h = 2^{-5}$	20	$5.8\,10^{-4}$	4	0.8	2
$h = 2^{-6}$	28	$3.3\,10^{-4}$	7	0.8	22
$h = 2^{-7}$	36	$2.0\,10^{-4}$	3	0.8	95

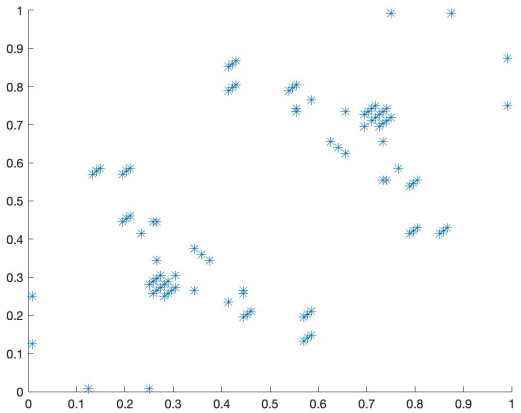

Figure 4: Active sets of the last iteration in case 2^7

The active set for $0.41h^{2/5}$ and $h = 2^{-7}$ is displayed in Figure 3. We observe that active sets are gathered close to the central circle and corners, where u is not smooth enough or has large derivatives.

4.3 C^1 Example

We consider the following f, which becomes unbounded near the corner $(1, 1)$ of Ω, and the corresponding exact solution u, which is twice differentiable in Ω but possesses an unbounded gradient near $(1, 1)$:

$$u(x) = -\sqrt{2 - |x|^2}, \quad f(x) = 2\left(2 - |x|^2\right)^{-2} \quad \forall x \in \Omega$$

we choose $\delta, \theta \approx h^{1/2}$. **Perron Iteration**:

Mesh size	number of points on circle	$L_\infty-$ error	Perron steps
$h = 2^{-4}$	20	$4.8\,10^{-3}$	39
$h = 2^{-5}$	28	$2.1\,10^{-3}$	52

Monotone Operator

Mesh size	number of points on circle	$L_\infty-$ error	Newton steps	Damping
$h = 2^{-4}$	20	$1.2\,10^{-2}$	13	0.8
$h = 2^{-5}$	28	$8.2\,10^{-3}$	13	0.8
$h = 2^{-6}$	44	$4.9\,10^{-3}$	13	0.8

5 Conclusion

- Compared to Newton iterations, Perron iterations are much slower. We may further investigate parallel ways to conduct this algorithm.
- For singular or non-smooth cases, Newton damping must be introduced to ensure convergence.
- Note that a goot initial value is crucial to Newton iteration. We may chose and modify them carefully.
- We observe almost linear convergence rates for monotone operator. Accurate operator and filter scheme significantly improve computational accuracy.
- We observe that filter scheme does't always have better accuracy over accurate operator. Active sets are close to locations where solutions are not very smooth or have relatively large derivatives.

References

[1] R. H. Nochetto, D. Ntogkas, and W. Zhang. Two-scale method for the Monge-Ampère equation: Convergence to the viscosity solution. *Mathematics of Computation*, 88(316):637–664, May 2018.

[2] Ricardo H. Nochetto and Dimitrios Ntogkas. Convergent filtered scheme for the Monge-Ampère Equation, July 2018. arXiv:1807.04866.