J Sci Comput (2016) 69:556–580
DOI 10.1007/s10915-016-0215-8

An Accelerated Method for Nonlinear Elliptic PDE

Hayden Schaeffer[1] · **Thomas Y. Hou**[2]

Received: 29 September 2015 / Revised: 5 April 2016 / Accepted: 27 April 2016 /
Published online: 12 May 2016

Abstract We propose two numerical methods for accelerating the convergence of the standard fixed point method associated with a nonlinear and/or degenerate elliptic partial differential equation. The first method is linearly stable, while the second is provably convergent in the viscosity solution sense. In practice, the methods converge at a nearly linear complexity in terms of the number of iterations required for convergence. The methods are easy to implement and do not require the construction or approximation of the Jacobian. Numerical examples are shown for Bellman's equation, Isaacs' equation, Pucci's equations, the Monge–Ampère equation, a variant of the infinity Laplacian, and a system of nonlinear equations.

Keywords Nonlinear elliptic PDE · Degenerate elliptic PDE · Accelerated convergence · Elliptic systems · Finite difference methods · Viscosity solutions · Fixed point methods

Mathematics Subject Classification 65B05 · 65N06 · 65M22 · 49L25 · 35J60 · 35B51 · 35J70

1 Introduction

Nonlinear and degenerate elliptic partial differential equations arise in many fields of mathematics, engineering, and science. In astrophysics, differential geometry, fluid dynamics, and imaging science nonlinear models typically provide better mathematical models to a wider range of phenomena; however, they can be challenging to analyze and simulate. In particular, even when it is possible to construct a convergent numerical approximation, it can be computational expensive to solve in practice. Therefore, many of the recent works in numerical

✉ Hayden Schaeffer
schaeffer@cmu.edu

1 Carnegie Mellon University, Pittsburgh, PA, USA

2 California Institute of Technology, Pasadena, CA 91125, USA

methods for nonlinear PDE focus on the construction of convergent and/or efficient schemes [1–22].

In this work, we are concerned with second order elliptic operators, which can be written in the form of $H(u) = \mathcal{H}(x, u, Du, D^2u)$. They can be nonlinear and/or degenerate elliptic operators, but for simplicity we will refer to this general class as elliptic. The analytic framework for the well-posedness of such equations tend to rely on the theory of viscosity solutions. A function $\mathcal{H}$ is (degenerate) elliptic if $\mathcal{H}(x, r, p, X) \leq \mathcal{H}(x, s, p, Y)$ whenever $r \leq s$ and $Y - X$ is positive semi-definite. Numerical schemes for elliptic problems converge to the correct viscosity solution provided they satisfy a discrete comparison principle (typically in the form of monotonicity) along with a uniform bound and consistency [23]. In some recent works [24,25], finite difference stencils are constructed using the theory of viscosity solutions in order to guarantee convergences to the correct solution. The discretization results in a system of nonlinear algebraic equations that may only be Lipschitz continuous; therefore, convergent implicit methods may be difficult and many times not possible to construct.

When applicable, Newton-based methods are preferred, since in the ideal case, the method can converge superlinearly. They have been successfully applied to nonlinear elliptic problems, for example in the works of [6,14,15,26]. However, the use of Newton-based methods require problem specific adaptation and are not directly applicable. In [26], the convergence of Newton's method for the Monge–Ampère equation is shown in the continuous setting, but not in the discrete sense. For non-monotone discretizations of second order nonlinear elliptic PDE, Newton's method can become unstable [15]. In [14], the use of monotone stencils results in a non-differentiable discrete scheme, and thus a degenerate Jacobian, which requires some regularization to ensure the convergence of Newton's method. Preconditioning was also used to avoid ill-conditioning of the problem for singular examples. As explained in [8], the convergence of Newton's method for nonlinear elliptic PDE found in [22] requires a good initial guess; however, the construction of an initial guess is not clear. In [27], it was stated that for numerical solutions to non-smooth examples, time marching methods are preferred over Newton's method.

Unlike Newton-based methods, for a general nonlinear equation, the standard discrete fixed point iteration will converge to the correct solution given very mild conditions. The standard discrete fixed point iteration is equivalent to embedding the elliptic equation into a parabolic equation:

$$\begin{aligned}\partial_t u(x,t) &= -H(u(x,t))\\ u(x,0) &= u_0\end{aligned}$$

where (formally) $u(x,t) \to u^*(x)$ as $t \to \infty$ and u^* is the unique steady state solution, that is $H(u^*) = 0$. Since the goal is to find the steady state of the time dependent solution $u(x,t)$ and not temporal accuracy, the forward Euler method is preferred. Typically, the time it takes to reach steady state follows an algebraic decay rate: $||u(x,t) - u^*(x)|| = \mathcal{O}(t^{-p})$, where p is a constant that depends on the problem. To reach time t with a fixed time step $\tau > 0$ requires n iterations satisfying $t = \tau n$. For second order equations, stability requires that the time step $\tau = \mathcal{O}(h^2)$, where $h > 0$ is the grid size. Therefore, to reach an error of ϵ requires $n = \mathcal{O}(h^{-2}\,\epsilon^{-p})$ iterations, which becomes increasingly worse as the grid is refined.

The standard fixed point iteration is also closely related to the gradient descent method used throughout the field of optimization. The gradient descent method is used to minimize a convex functional $F(u)$ by descending in the negative direction of its first variation, i.e. the gradient flow

$$\partial_t u(x,t) = -\nabla_u F(u(x,t)).$$

The method is easy to implement and is convergent; however, it suffers from a slow convergence rate. In terms of the functional, the gradient descent method decreases the energy as:

$$F(u(-,t)) - F(u^*) = \mathcal{O}(t^{-1}).$$

To accelerate convergence towards the minimizer, several new methods have been proposed, which are related to the work of [28]. By extrapolating forward using previous iterations, the accelerated gradient descent method in [28] has the optimal convergence rate of

$$F(u(-,t)) - F(u^*) = \mathcal{O}(t^{-2})$$

for any convex minimization problem whose first variation is Lipschitz.

Inspired by the technique in [28,29], we construct an extrapolation method that is shown in practice to dramatically accelerate fixed point methods for elliptic PDEs. Unlike the work in optimization, we assume that H is a nonlinear elliptic operator but do not assume a variational structure to the PDE, i.e. H may not be the first variation of a functional. Also, unlike the Newton and quasi-Newton methods, we construct a nonlinear algebraic solver which does not require the construction or approximation of the Jacobian (or higher order derivatives).

In this work, we propose two extrapolation methods. We show that the first method is linearly stable and, if convergent, would converge quadratically fast to the steady state solution. The second method is provably convergent and in practice has a convergence rate similarly to the first. In the numerical experiments, we see both methods converge to the steady state faster than the standard fixed point methods with a lower complexity in terms of the grid size.

The outline of this work is as follows. In Sect. 2, we review some previous work in viscosity solutions and numerical convergence. We present our two methods and algorithms in Sect. 3. We analyze our numerical methods in Sects. 4, 5. In Sect. 6, numerical results are provided for one and two dimensional examples, detailing the experimental convergence rates, complexity and applicability to various elliptic equations. Concluding remarks are given in Sect. 7.

2 Preliminary Remarks and Notation

Before introducing our work, we give a brief overview of the related theory and our notation. Let

$$H[u]:=\mathcal{H}(x, u, Du, D^2u)$$

be a degenerate elliptic operator. We say that u is a supersolution if $H[u] \geq 0$ and u is a subsolution if $H[u] \leq 0$. We call u the (viscosity) solution if it is both a super- and subsolution. For the rest of this work we implicitly assume that for a given H we have the existence of a unique (viscosity) solution u^*. To construct the numerical solutions, we discretize the operator and solve the resulting equation, $H_h(u) = 0$, for a fixed grid size $h > 0$. We call u_h the unique solution for the discrete problem (the subscript may be dropped at times for simplicity).

If H_h is a monotone, stable and consistent approximation and H satisfies a comparison principle, then by Theorem 2.1 of [23] we have that as $h \to 0$, u_h converges locally uniformly to the unique continuous viscosity solution of $H[u] = 0$. This provides sufficient conditions for a numerical scheme to converge in the viscosity sense. Several recent works focus on the construction of particular numerical schemes which obey these conditions, as well as the analysis of their convergence rates in terms of h. For example, in [25], monotone stencils for fully nonlinear elliptic equations of the form $H(D^2u) = 0$ are constructed which have algebraic rates of convergence. In [24], *degenerate elliptic finite difference stencils* were introduced, which are shown to be convergent approximations to general degenerate elliptic problems. The degenerate elliptic finite difference scheme satisfies a discrete comparison principle which is used to show that the forward Euler method applied to degenerate elliptic schemes converges to the unique solution.

For a general monotone approximation to a nonlinear elliptic operator, the Euler map is guaranteed to converge for any bounded initial guess and is simple to implement directly. However, the number of iterations necessary to converge to a solution of $H_h(u) = 0$ grows with the grid size, thereby making it computationally inefficient for fine grids. Motivated by [28], our approach is to accelerate the computation of the discrete nonlinear algebraic equation using the viscosity solution theory from above. In particular, we use the notion of discrete viscosity solutions to construct a numerical method which approaches the solution from above (or below) in an accelerated manner.

3 Acceleration Methods

In this section, we present two algorithms for finding the unique solution to $H_h(u) = 0$. The first is a two-step algorithm which uses prior iterates to move solutions further toward steady state. The second method is a super/subsolution-preserving version of the first method. In order to maintain consistent contraction toward the stationary solution, a decision is made whether to use the given extrapolated iterate or to update using a forward Euler step.

3.1 Method 1: Direct Extrapolation

The first algorithm uses successive iterates in order to continue the solution toward the steady state. The first step is to linearly extrapolate forward “in time” using the previous two iterates. Given u^n and u^{n-1} and a parameter γ_n, the extrapolation is calculated by:

$$u^{n,E} = u^n + \gamma_n(u^n - u^{n-1}).$$

For the rest of this work, we will assume that we have a sequence of parameters $\gamma_n \in [0, 1)$ which will determine speed of the extrapolation. In practice, we have found that the choice of $\gamma_n = \frac{n}{n+n_0}$ for some $n_0 \geq 10$ is sufficient. The second step applies a forward Euler step to the extrapolated values. The method is summarized in Algorithm 1.

```
Algorithm 1
Given: $u^0$, $u^1 := u^0 - \tau H(u^0)$, and parameters $\{\gamma_n\}$
while $||u^n - u^{n-1}||_\infty > tol$ do
    $u^{n,E} = u^n + \gamma_n(u^n - u^{n-1})$
    $u^{n+1} = u^{n,E} - \tau H(u^{n,E})$
end while
```

Remark 3.1 The extrapolation resembles the sequence generated by secant-like methods (1d):

$$u_{secant} = u^n - H(u^n)\frac{u^n - u^{n-1}}{H(u^n) - H(u^{n-1})}$$

in which the function evaluations are replaced by a sequence of scalars. As mentioned before, H may not be regular enough for Newton and secant-based methods to be directly applicable.

However, comparing the methods provides some insight into the behavior of our algorithm. In particular, we see that the extrapolation is more efficient since it does not require additional function evaluations.

3.2 Method 2: Super/Subsolution Preserving

When the iterates are far from the steady state solution and the difference between them are relatively small, in particular $||u^n - u^{n-1}||_\infty < \gamma_n^{-1}||u^n - u^*||_\infty$, then the extrapolation step accelerates the iterations closer to the steady state solution. On the other hand, when solution are close to steady state, the extrapolation may move past u^*, causing the method to oscillate around the steady state. To control for this, the second algorithm requires that the extrapolation preserves the sign of the function H and the sign of the difference between iterations.

The idea is to construct a decreasing (increasing) sequence of supersolutions (subsolutions) which converge to the steady state solution. When the iterates are ordered properly, then the extrapolation is contractive, otherwise a forward Euler step is preferred.

The algorithm is summarized below.

```
Algorithm 2 (Supersolution Case)
Given: $u^0$ and $u^1 := u^0 - \tau H(u^0)$, $u^1 \leq u^0$, $H(u^0) \geq 0$ and parameters $\{\gamma_n\}$
while $||u^n - u^{n-1}||_\infty > tol$ do
    $u^{n,E} = u^n + \gamma_n(u^n - u^{n-1})$
    if $u^n \leq u^{n-1}$ and $H(u^{n,E}) > 0$ then
        $u^{n+1} = u^{n,E} - \tau H(u^{n,E})$
    else
        $u^{n+1} = u^n - \tau H(u^n)$
    end if
end while
```

The same interpretation holds for the subsolution case. The initialization for Algorithm 2 requires that the iterations are properly ordered, which is satisfied by taken a forward Euler step.

```
Algorithm 2 (Subsolution Case)
Given: $u^0$ and $u^1 := u^0 - \tau H(u^0)$, $H(u^0) \leq 0$ and parameters $\{\gamma_n\}$
while $||u^n - u^{n-1}||_\infty > tol$ do
    $u^{n,E} = u^n + \gamma_n(u^n - u^{n-1})$
    if $u^n \geq u^{n-1}$ and $H(u^{n,E}) < 0$ then
        $u^{n+1} = u^{n,E} - \tau H(u^{n,E})$
    else
        $u^{n+1} = u^n - \tau H(u^n)$
    end if
end while
```

In both cases, Method 2 can have the same complexity and speed as Algorithm 1 and at worst the speed of the forward Euler method with twice the number of function evaluations. In Sect. 6, we show that on the examples we tested, the convergence rate for Method 2 was faster than the forward Euler method with lower complexity.

Remark 3.2 In the supersolution formulation, since $H(u^n)$ is positive and decreasing in n, our extrapolation method and the secant-like methods move in the same direction. This is also true of the subsolution version.

4 Analysis of Method 1

One way to view the iterates, $\{u^n\}$, generated by Method 1 is as an approximation to the following redundant system of parabolic equations:

$$\begin{cases} u_t &= u_t \\ u_t &= -H(u) \end{cases} \tag{4.1}$$

Using this PDE interpretation we show that if the method is convergent, then $u^n \approx u(t_n)$ where $t_n = \mathcal{O}(n^2)$. Although the convergence of Method 1 is not guaranteed for general second order equation, we can show that for a certain class it is linearly stable.

4.1 Temporal Consistency

Given a sequence of parameters $\{\gamma_n\}$, define the sequence α_n by:

$$\begin{cases} \alpha_1 &= \gamma_1 \\ \alpha_j &= (1+\alpha_{j-1})\gamma_j, \text{ for } j > 1 \end{cases} \tag{4.2}$$

We will show that if the sequence $\{u^n\}$ generated by Method 1 is a convergent approximation of Eq. (4.1), then u^n is consistent with $u(\tau(n + \sum_{j=1}^{n-1} \alpha_j))$. Note that the standard forward Euler method approximates the sequence $u(\tau n)$.

Proposition 4.1 (Temporal Truncation Error for Method 1) *Let $\{u^n\}$ be the sequence generated by the Method 1 and let $||u_{tt}||_{L^\infty} \le C < \infty$. Then the scheme is consistent with $\{u(t_n)\}$ where $t_n := \tau(n + \sum_{j=1}^{n-1} \alpha_j)$.*

Proof First, we wish to show that the linear extrapolation step is consistent with $u_t = u_t$. When written in finite difference form, if $t_{n,E} := t_n + \gamma_n(t_n - t_{n-1})$, we can see that consistency holds:

$$\frac{u^{n,E} - u^n}{t_{n,E} - t_n} - \frac{u^n - u^{n-1}}{t_n - t_{n-1}} = \frac{u^{n,E} - u^n}{\gamma_n(t_n - t_{n-1})} - \frac{u^n - u^{n-1}}{t_n - t_{n-1}} = 0 \tag{4.3}$$

where the second equality is exactly the extrapolation step.

More precisely, for any u which is twice continuously differentiable with respect to t, we have the following two Taylor expansions:

$$\begin{aligned} u^{n-1} &= u^n - u_t^n \, \Delta t_n + u_{tt}(\eta_{n,1}) \, \Delta t_n^2 \\ u^{n,E} &= u^n + u_t^n \, \gamma_n \Delta t_n + u_{tt}(\eta_{n,2}) \, \gamma_n^2 \Delta t_n^2 \end{aligned}$$

where $\eta_{n,1} \in (t_{n-1}, t_n)$, $\eta_{n,2} \in (t_n, t_{n,E})$ and $\Delta t_n := t_n - t_{n-1}$. The local temporal truncation error incurred in the extrapolation step is given by:

$$\begin{aligned} T_e^n &= u^{n,E} - u^n - \gamma_n(u^n - u^{n-1}) \\ &= \gamma_n u_t^n \, \Delta t_n - \gamma_n u_t^n \, \Delta t_n + u_{tt}(\eta_{n,2}) \, \gamma_n^2 \Delta t_n^2 - u_{tt}(\eta_{n,1}) \, \gamma_n \Delta t_n^2 \\ &= u_{tt}(\eta_{n,2}) \, \gamma_n^2 \Delta t_n^2 - u_{tt}(\eta_{n,1}) \, \gamma_n \Delta t_n^2. \end{aligned}$$

Note that using the sequence α_n, the time step can be simplified as:

$$\Delta t_n = t_n - t_{n-1} = \tau \left(\left(n + \sum_{j=1}^{n-1} \alpha_j \right) - \left(n - 1 + \sum_{j=1}^{n-2} \alpha_j \right) \right) = \tau(1 + \alpha_{n-1}).$$

and the extrapolated time stamp can be written as $t_{n,E} = \tau(n + \sum_{j=1}^{n} \alpha_j)$. Furthermore, we can also simplify the truncation error of the extrapolation step to:

$$\begin{aligned} T_e^n &= u_{tt}(\eta_{n,2})\, \gamma_n^2 \tau^2 (1+\alpha_{n-1})^2 - u_{tt}(\eta_{n,1})\, \gamma_n \tau^2 (1+\alpha_{n-1})^2 \\ &= u_{tt}(\eta_{n,2})\, \tau^2 \alpha_n^2 - u_{tt}(\eta_{n,1})\, \tau^2 \alpha_n (1+\alpha_{n-1}). \end{aligned}$$

Next, the update step in finite difference form is given by:

$$\frac{u^{n+1} - u^{n,E}}{\tau} + H(u^{n,E}) = 0.$$

We assume that $H(u^{n,E})$ is given exactly, since we are only concerned with temporal consistency. Next, expanding around the extrapolation iterate yields:

$$u^{n+1} = u^{n,E} + u_t^{n,E}\tau + u_{tt}(\eta_n)\frac{\tau^2}{2}$$

where $\eta_n \in (t_{n-1,E}\, , t_n)$ and $t_{n+1} - t_{n,E} = \tau$. The local temporal truncation error for the updated step is given by:

$$T_{FE}^{n+1} = u_{tt}(\eta_n)\frac{\tau^2}{2}$$

Altogether, the truncation error is $\mathcal{O}(\tau^2)$ depending on the second derivative; thus, for any C^2 function (in time) as $\tau \to 0$, the local truncation error goes to zero. □

The overall temporal accuracy of the method is on the same order as the forward Euler method. For steady state problems, the time-step τ is fixed since u_{tt} is assumed to have sufficient decay.

The time stamps t_n depend on the sequences γ_n and α_n. With mild assumptions on the growth of γ_n, the following proposition provides lower bounds for the growth rate of α_n, and in turn t_n.

Proposition 4.2 *Let γ_n be a non-negative sequence with $1 - \gamma_n \leq \frac{1}{n+1}$ and let α_n be defined by the sequence generated as follows:*

$$\begin{cases} \alpha_1 &= \gamma_1 \\ \alpha_j &= (1+\alpha_{j-1})\gamma_j, \textit{ for } \ j > 1 \end{cases}$$

and t_n to be the time stamps of Method 1, then $\alpha_n \geq \frac{n}{2}$ and $t_n \geq \tau\frac{n^2-n}{4}$.

Thus if the method is convergent, it should converge quadratically faster than forward Euler. For general nonlinear $H(u)$, the arguments in [30] can be used to guarantee a convergence proof; however, it would require $\tau = \mathcal{O}(h^4)$. This severe time step restriction is a result of the quadratic rate of growth of t_n. The theory provided in [30] is more general and does not account for the ellipticity properties of $H(u)$.

4.2 Linear Stability Analysis

Consider the linearization of H which is uniformly elliptic, in particular let $H(u) = Au$, where A is positive definite. With this form, Method 1 becomes a linear explicit two-step method:

$$\begin{aligned} u^{n+1} &= u^n + \gamma_n(u^n - u^{n-1}) - \tau H(u^n + \gamma_n(u^n - u^{n-1})) \\ &= (1+\gamma_n)(1-\tau A)u^n - \gamma_n(1-\tau A)u^{n-1} \end{aligned}$$

Next, the equation for u^n is transformed to a system of decoupled vectors w_k^n in the eigenbasis:

$$w_k^{n+1} = (1+\gamma_n)(1-\lambda_k\tau)w_k^n - \gamma_n(1-\lambda_k\tau)w_k^{n-1}$$

where $\lambda_k > 0$ are the corresponding eigenvalues of A. For each vector w_k^n, the updates can be written as a two-step linear system with

$$\begin{bmatrix} w_k^n \\ w_k^{n+1} \end{bmatrix} = B_k^n \begin{bmatrix} w_k^{n-1} \\ w_k^n \end{bmatrix} = \left(\prod_{j=1}^{n} B_k^j\right) \begin{bmatrix} w_k^0 \\ w_k^1 \end{bmatrix}$$

where

$$B_k^n := \begin{bmatrix} 0 & 1 \\ -\gamma_n(1-\lambda_k\tau) & (1+\gamma_n)(1-\lambda_k\tau) \end{bmatrix}$$

If $\lambda_k\tau < 1$ hold for every k, then

$$\left\| \prod_{j=1}^{n} B_k^j \right\| \to 0 \quad \text{as} \quad n \to \infty.$$

Therefore, for any bounded u^0 and u^1, the solution $u^n \to 0$ as $n \to \infty$. Although this does not guarantee the convergence of the method for degenerate H, we see that this analysis gives local convergence for nonlinear H with uniformly elliptic linearization near u^*.

5 Analysis of Method 2

Method 2 can be written in the form of Method 1 by defining extrapolation parameters $\{\bar{\gamma}_n\}$:

$$\bar{\gamma}_n = \begin{cases} \gamma_n, & \text{if } u^n \le u^{n-1} \text{ and } H(u^{n,E}) > 0 \\ 0, & \text{otherwise.} \end{cases}$$

(for the supersolution case) and the sequence $\bar{\alpha}_n$ by

$$\begin{cases} \bar{\alpha}_1 &= \bar{\gamma}_1 \\ \bar{\alpha}_j &= (1+\bar{\alpha}_{j-1})\bar{\gamma}_j, \ \text{for} \ j > 1 \end{cases} \tag{5.1}$$

Proposition 5.1 (Temporal Truncation for Method 2) *Let* $\{u^n\}$ *be the sequence generated by Method 2 and let* $||u_{tt}||_{L^\infty} \le C < \infty$. *Then the scheme is consistent with* $\{u(t_n)\}$ *where* $t_n := \tau(n + \sum_{j=1}^{n-1} \bar{\alpha}_j)$.

The proof of the proposition above follows from the proof of Proposition 4.1, by ignoring the extrapolation step when $\bar{\gamma}_n = 0$. If the scheme is convergent we see that the rate at which it converges can be at worst the rate of forward Euler and at best quadratically faster than forward Euler. We will show in Sect. 6 that in practice the algorithm has a convergence rate comparable to (or even better than) Method 1.

The update condition in Method 2 is based on the idea of approaching the discrete solution from above (or below) and relies on a discrete generalization of the comparison principle.

Definition 5.2 (*Discrete comparison principle*) Let H_h be an operator that has the comparison principle for $h > 0$. If u satisfies $H_h(u) \ge 0$ ($H_h(u) \le 0$), u is said to be a discrete supersolution (subsolution) of H_h, *i.e.* $u \ge u_h^*$ ($u \le u_h^*$) where u_h^* is the unique solution to $H_h(u_h^*) = 0$.

In addition to the comparison principle, the convergence of Method 2 relies on a well-behaved forward Euler map as well as the existence of a non-trivial time step. The definition below will be taken as an assumption throughout this section.

Definition 5.3 (*Forward Euler*) Let H_h have a finite Lipschitz constant K and satisfy the discrete comparison principle. The forward Euler map, $T_{FE}u{:=}u - \tau H_h(u)$, is defined with optimal time step $\tau_{op} > 0$ if for any $\tau_{op} \geq \tau > 0$ the following hold:

- T_{FE} is monotone,
- For any u and v, there exists constant C, with $0 \leq C < 1$, such that $||T_{FE}u - T_{FE}v|| \leq C||u - v||$.

The assumptions above are commonly used in the construction of monotone Runge–Kutta methods and are related to strongly stable schemes [30–34]. Other related conditions exists, for example *positive type* scheme are used in [35], to construct consistent finite difference stencils for nonlinear second order equations. Also, the degenerate elliptic schemes in [24] are monotone and thus satisfy the discrete comparison principle.

If H_h is consistent, then forward Euler iterations will converge to the correct solution in the limit $h \to 0$. The second property in Definition 5.3 requires that the forward Euler is a strict contraction. In practice, it is not always possible to find a C uniformly for all u and v. For example if H is differentiable with zero derivative at u^*, then the constant $C_n = ||I - \tau DH(u^n)||$ goes to 1 as $u^n \to u^*$. So the second requirement can be weakened to having the product of the constants C_n going to zero as $n \to \infty$. Without loss of generality, we will assume that one can choose a uniform contractive constant C.

The following theorem shows that as long as the forward Euler map is well behaved, then the Method 2 is convergent.

Theorem 5.4 (Convergence of Method 2) *Let $\{u_h^n\}$ be the sequence generated by Method 2 for a fixed space step $h > 0$ and H_h is a convergent discretization of H with Lipschitz constant K. Then with time step restriction $\tau K < 1$ the following hold*

- *If u_h^0 is a supersolution (subsolution), then $\{u_h^n\}$ is a supersolution (subsolution) with respect to each u_h^*.*
- *For fixed h, $||u_h^n - u_h^*||_\infty \to 0$ monotonically in n.*
- *And $u_h^n \to u^*$ as $n \to \infty$ and $h \to 0$*

Proof With the time step $\tau K \leq 1$ the forward Euler update $T_{FE}u{:=}u - \tau H(u)$ is a monotone scheme, i.e.

$$\text{if } \ w \geq v \ \text{ then } \ T_{FE}w \geq T_{FE}v$$

In particular, by taking $v = u_h^*$ (which is the fixed point) and a supersolution w^0, then w^n is a supersolution for all n. Thus the forward Euler map preserves discrete supersolutions.

For any iteration, Method 2 has two possible updates. First, if no extrapolation is used, then for τ satisfying $\tau K \leq 1$, the forward Euler method is contractive in the maximum norm, i.e.

$$||u^{n+1} - u^*||_\infty \leq C||u^n - u^*||_\infty \tag{5.2}$$

for some $0 \leq C < 1$.

In the second case, if the extrapolation step holds, then the iterates must satisfy the following two conditions: $u^n \leq u^{n-1}$ and $u^{n,E}$ is a discrete supersolution. Therefore, the extrapolation step does not increase the distance to the solution:

$$0 \leq u^{n,E} - u^* = u^n + \gamma_n(u^n - u^{n-1}) - u^* \leq u^n - u^*$$

Thus the update is a discrete supersolution and contracts towards the steady state solution with the following estimate:

$$||u^{n+1} - u^*||_\infty \leq C||u^{n,E} - u^*||_\infty \leq C||u^n - u^*||_\infty$$

In both cases, the distance to the steady state solution is decreasing:

$$||u^{n+1} - u^*||_\infty \leq C||u^n - u^*||_\infty$$

The above arguments hold for subsolutions as well, by reversing some of the inequalities.

Lastly, the error between the iterates and the solution to the continuous problem is bounded by:

$$||u_h^n - u^*||_\infty \leq ||u_h^n - u_h^*||_\infty + ||u_h^* - u^*||_\infty \tag{5.3}$$

Since H_h is a convergent scheme then $u_h^* \to u^*$ as $h \to 0$, so we have $u_h^n \to u^*$ as $n \to \infty$ and $h \to 0$. □

Remark 5.5 The convergence rate is determined by the accuracy of the discretization of H_h. In this work, we assume that we are given an accurate discretization and are concerned with the iterative convergence at a fixed resolution $h > 0$.

Remark 5.6 If the scheme for H_h is of order h^r, then only partial convergence of the iterative scheme is needed, i.e.

$$||u_h^n - u_h^*||_\infty \leq Ch^r,$$

to preserve the spatial convergence rate for the method.

6 Computational Examples

In this section, we apply both numerical methods as well as the forward Euler method to various elliptic PDE. In each case we take a uniform discretization of the square $\Omega = (-1, 1)^2$ with the spatial step size defined as $h{:=}\frac{2}{N}$, with N indicating the number of points in either dimension. Also, in each example there is specified Dirichlet boundary conditions given by the function $g(x, y)$. The parameter γ_n is given by $\gamma_n = \frac{n}{n+C}$ for some $C > 1$. In most cases, we have used the exact solution so that we are able to compare the two numerical methods directly.

6.1 Bellman's Equation

Bellman's equation is a fully nonlinear uniformly elliptic convex PDE defined as:

$$\sup_{a\in\mathcal{A}} L_a u = f$$

where L_a are parameterized linear operators. In general, any fully nonlinear uniformly elliptic convex PDE can be written in this form. In our numerical experiments, we take the following two examples:

$$\max(u_{xx}, (1 + x^2)u_{xx}) = x^3 + x$$
$$u(-1) = u(1) = 0$$

whose exact solution is $u(x) = \frac{1}{6}x^3 + \frac{1}{40} - \frac{23}{120}x + \frac{1}{20}x^4 \min(x, 0)$ and

$$\max(u_{xx}, 3u_{xx}) = \text{sign}(x)|x|^{1/3}$$
$$u(-1) = u(1) = 0$$

whose exact solution is $u(x) = \frac{9}{28}\left(-|x|^{4/3}\max(-x, -3x) - 2x + 1\right)$. These two examples are chosen for their different regularities. The first has a C^2 solution where as the second is only $C^{1,\frac{1}{3}}$. To discretize u_{xx}, we take the standard second order stencil:

$$(u_i)_{xx} = \frac{u_{i+1} - 2u_i - u_{i-1}}{h^2}$$

where u_i is the approximation of $u(x_i)$. The resulting approximation to the Bellman operator satisfies the comparison principle.

In Fig. 1, the error and solution plots of the first example are shown for the two proposed methods and the forward Euler method. The solutions are computed on a 2048 size grid with a time step parameter of $\tau = \frac{h^2}{8}$. The initial data is taken to be $u_0(x) = 1 - x^2$, which is a supersolution. Figure 1 (left) shows the error (on a log scale) versus the number of iterations for all three methods. The oscillatory nature of Method 1 around the steady state can be seen in the descending bumps. Although Method 1 reaches a smaller error as compared to Method 2 at various intermediate times, Method 2 is able to reach steady state faster.

Similarly, in Fig. 2, the error and solution plots of the second example are shown for all three methods. The solutions are computed on a 2048 size grid with a time step parameter of $\tau = \frac{h^2}{3}$. The initial data is taken to be $u_0(x) = x^2 - 1$, which is a subsolution. Figure 1 (left) shows the error (on a log scale) versus the number of iterations for all three methods. The cusps that appear in the error curve for Method 2 correspond to the method switching to a forward Euler step.

In Table 1, we study the iteration complexity of each method applied to the C^2 example of Bellman's equation from Fig. 1. The spatial error is the same for all methods, since the stencil H_h is the same, and has a numerical accuracy of about $h^{2.01}$. The column titles are

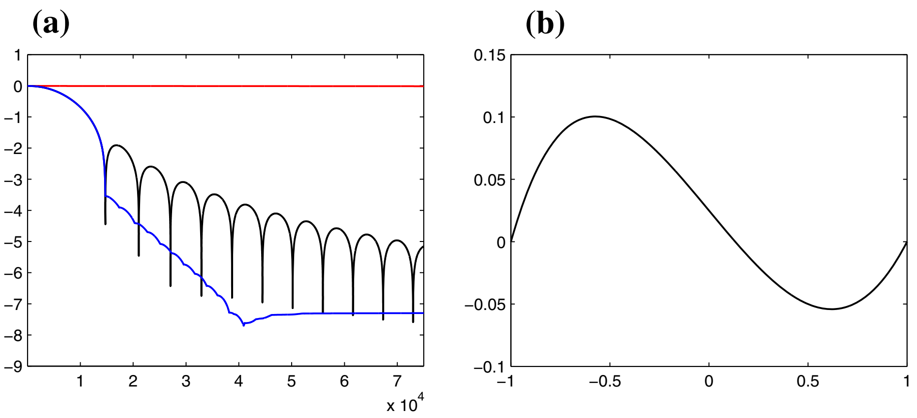


Fig. 1 Error and solution plots of the Bellman equation on a 2048 size grid. All three methods have the same initial conditions and time step parameter $\tau = \frac{h^2}{8}$. The first plot (*left*) shows the error (on a log scale) versus the number of iterations for the forward Euler method (*red*), Method 1 (*black*) and Method 2 (*blue*). **a** Error plot, **b** solution (Color figure online)

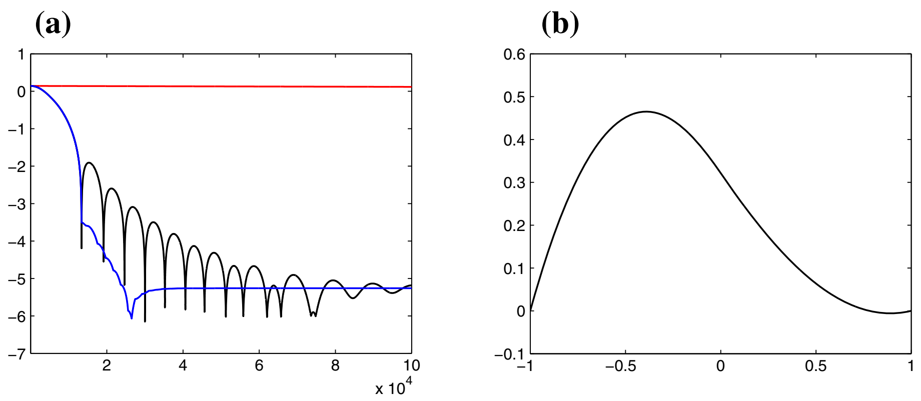


Fig. 2 Error and solution plots of the Bellman equation on a 2048 size grid. All three methods have the same initial conditions and time step parameter $\tau = \frac{h^2}{3}$. The first plot (*left*) shows the error (on a log scale) versus the number of iterations for the forward Euler method (*red*), Method 1 (*black*) and Method 2 (*blue*). **a** Error plot, **b** solution (Color figure online)

Table 1 Convergence comparison between various methods

Grid size	Error	M1 (SS)	M1 (LE)	M2	FE
128	1.3×10^{-5}	4757	1299	1957	144,744
256	3.3×10^{-6}	18,766	3349	5668	789,657
512	8.1×10^{-7}	50,228	6737	11,738	3,255,780
1024	2.0×10^{-7}	106,220	25,021	23,456	–
2048	5.1×10^{-8}	248,512	67,329	40,522	–
Rate	2.01	1.39	1.43	1.08	2.25

Using the C^2 solution of the Bellman equation from Fig. 1, we compare the complexity in terms of grid size and the number of iterations require to converge between the methods. The column titles are as follows: M1 (SS) is the number of iterations it takes for Method 1 to converge to steady state, M1 (LE) is the number of iterations it takes for Method 1 to reach the desired error, M2 is the number of iterations it takes for Method 2 to converge to steady state, and FE is is the number of iterations it takes for forward Euler to converge to steady state

as follows: M1 (SS) is the number of iterations it takes for Method 1 to converge to steady state, M1 (LE) is the number of iterations it takes for Method 1 to reach the desired error, M2 is the number of iterations it takes for Method 2 to converge to steady state, and FE is is the number of iterations it takes for forward Euler to converge to steady state. Method 2 has a nearly linear complexity while Method 1 is superlinear and forward Euler is quadratic. The quadratic complexity of forward Euler makes it difficult to calculate a numerical steady state solution on refined grids, even with very close initializations.

In Table 2, we study the iteration complexity of each method applied to the $C^{1,\frac{1}{3}}$ example of Bellman's equation from Fig. 2, with $\tau = \frac{h^2}{3}$. In this case, the stencil H_h has a numerical accuracy of about $h^{1.33}$ due to the regularity. Again, Method 2 has a nearly linear complexity even for non-regular solutions, while Method 1 is superlinear and forward Euler is quadratic.

Table 2 Convergence comparison between various methods

Grid size	Error	M1 (SS)	M1 (LE)	M2	FE
128	2.2×10^{-4}	3061	827	1424	107,066
256	8.7×10^{-5}	9992	1668	3771	527,278
512	3.5×10^{-5}	27,703	3351	8514	2,031,064
1024	1.4×10^{-5}	45,208	12,346	17,145	8,293,445
2048	5.5×10^{-6}	132,040	24,729	38,638	–
Rate	1.33	1.31	1.27	1.17	2.08

Using the $C^{1,1/3}$ solution of the Bellman equation from Fig. 2, we compare the complexity in terms of grid size and the number of iterations require to converge between the methods. M1 (SS) is the number of iterations it takes for Method 1 to converge to steady state, M1 (LE) is the number of iterations it takes for Method 1 to reach the desired error, M2 is the number of iterations it takes for Method 2 to converge to steady state, and FE is is the number of iterations it takes for forward Euler to converge to steady state

6.2 Isaacs' Equation

Isaacs' equations form a class of nonconvex fully nonlinear uniformly elliptic equations defined as:

$$\sup_{a\in\mathcal{A}} \inf_{b\in\mathcal{B}} L_{a,b}u = f$$

where $L_{a,b}$ are parameterized linear operators. The equation can be derived from particular two-player zero sum games [36], and represents a prototypical nonlinear nonconvex elliptic equation. Any fully nonlinear uniformly elliptic nonconvex PDE which is Lipschitz in all parameters can be written in this form [37]. For our example, we take the 3-term problem:

$$\max(L_3u, \min(L_1u, L_2u)) = x^2 + y^2$$

where the three linear operators are defined as:

$$\begin{aligned} L_1u &= (1+x^2)u_{xx} + (1+y^2)u_{yy} \\ L_2u &= u_{xx} + \frac{3}{2}u_{yy} \\ L_3u &= u_{xx} + u_{yy} \end{aligned}$$

with boundary data $g(x, y) = \max(x^4 - y^4, x^2 + y^3)$. For the discretization of the spatial derivatives, we take the standard second order stencils:

$$\begin{aligned} (u_{i,j})_{xx} &= \frac{u_{i+1,j} - 2u_{i,j} - u_{i-1,j}}{h^2} \\ (u_{i,j})_{yy} &= \frac{u_{i,j+1} - 2u_{i,j} - u_{i,j-1}}{h^2} \end{aligned}$$

where $u_{i,j}$ is the approximation of $u(x_i, y_j)$. The resulting approximation to Isaacs' equation satisfies comparison principle. For more on the theoretical behavior of Issacs' equation, see for example [37,38].

To calculate the error, we compute the solution using the standard forward Euler update on a 512-by-512 size grid and linearly interpolate onto the 256-by-256 size grid. The initial data, $u^0(x, y)$, is a solution of:

$$\Delta u^0(x, y) = x^2 + y^2$$

with the same boundary condition of the problem. This is a simple way to generate a subsolution for this case. The solution agrees with the refined grid calculations and we see that the convergence rate between our method are nearly identical (see Fig. 3).

6.3 Infinity Laplacian Based Equations

The infinity Laplacian is a degenerate quasi-linear PDE defined as:

$$\Delta_\infty u = \frac{1}{|\nabla u|^2} \sum_{i,j=1}^{n} u_{x_i} u_{x_j} u_{x_i x_j}$$

This PDE is related to the first variation of L^∞ functionals and has seen a surge of interest in various forms [39–44]. In particular, solutions are related to zero-sum differential games and can be interpreted as the expected payoff to a certain game (namely the *tug-of-war* games).

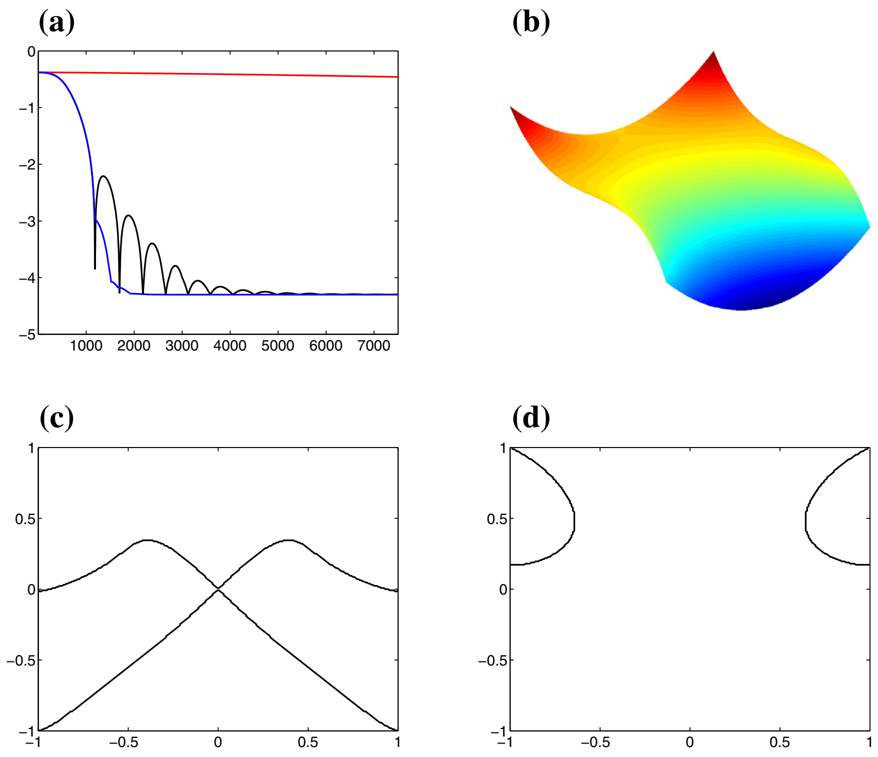


Fig. 3 Error and solution plots of the Isaacs equation on a 256-by-256 size grid. All three methods have the same initial conditions and time step parameter $\tau = \frac{h^2}{8}$. The first plot (*left*) shows the error (on a log scale) versus the number of iterations for the forward Euler method (*red*), Method 1 (*black*) and Method 2 (*blue*). In this case, the error in Method 1 and Method 2 agrees. In plots (**c**) and (**d**) the level curves between the set in which one operator determines the local solution is shown. **a** Error, **b** solution, **c** the level curves of the set $\{L_1 u \leq L_2 u\}$, **d** the level curves of the set $\{\min(L_1 u, L_2 u) \leq L_3 u\}$ (Color figure online)

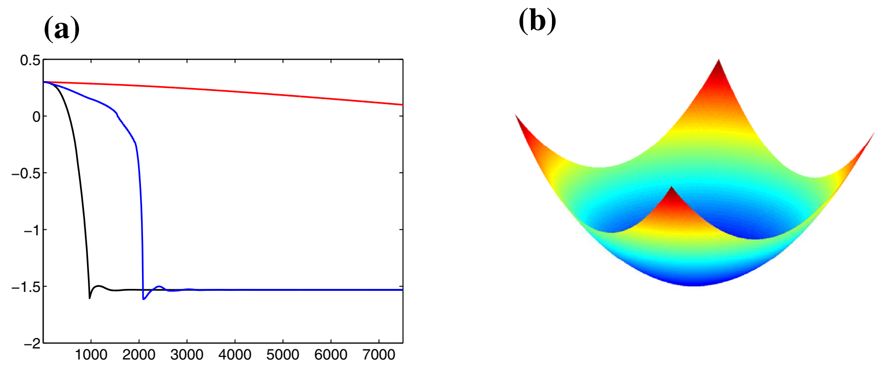


Fig. 4 Error and solution plots of the nonlinear differential game on a 128-by-128 size grid. All three methods have the same initial conditions and time step parameter $\tau = \frac{h^2}{4}$. The first plot (*left*) shows the error (on a log scale) versus the number of iterations for the forward Euler method (*red*), Method 1 (*black*) and Method 2 (*blue*). **a** Error, **b** solution (Color figure online)

Using the zero-sum game, the operator Δ_∞ has the following discrete form:

$$\Delta_\infty u = \frac{1}{\epsilon^2}\left(\max_{y\in\partial B(x,\epsilon)} u^\epsilon(y) + \min_{y\in\partial B(x,\epsilon)} u^\epsilon(y) - 2u^\epsilon(x)\right)$$

where $\epsilon = \mathcal{O}(h)$. There are a few convergent numerical methods that have been proposed that use this approach [1,3,4] to construct finite difference methods. It is worth noting that there are also finite element approximations to the infinity Laplacian using the vanishing moment method [6] and an adaptive method [11].

For our experiment we use a non-homogenous example found in [39]:

$$\Delta_\infty u + |\nabla u| = f$$

which has the following dynamic program:

$$u^\epsilon(x) - \rho \max_{y\in\partial B(x,\epsilon)} \left\{u^\epsilon(y) - \epsilon f(x)\right\} - (1-\rho) \min_{y\in\partial B(x,\epsilon)} \left\{u^\epsilon(y) + \epsilon f(x)\right\} = 0$$

with constant $\rho := \frac{2+\epsilon}{4}$. This provides the following numerical fixed point:

$$u^{n+1}_{i,j} = u^n_{i,j} - \frac{\tau}{(Ch)^2}\Big(u^n_{i,j} - \rho \max_{y\in\partial B(x,Ch)} \left\{u^n(y) - Chf(x)\right\}$$
$$- (1-\rho) \min_{y\in\partial B(x,Ch)} \left\{u^n(y) + Chf(x)\right\}\Big)$$

for some constant $C > 0$. By choosing the forcing term as $f = 1 + \sqrt{x^2+y^2}$ and the boundary data as $g = x^2 + y^2$, we have the exact solution $u = x^2 + y^2$ to this problem. The initial data is assigned as the maximum of the boundary data and is thus a supersolution. As seen in Fig. 4, for this degenerate equation Method 1 and Method 2 reaches state at roughly the same time; however, for intermediate times Method 2 may "stall".

6.4 Pucci's Equation

For the remaining examples, we consider fully nonlinear second order PDE which are functions of the eigenvalues of the Hessian in two dimensions. In particular, with the eigenvalue ordering $\lambda_1(D^2u(x)) \leq \lambda_2(D^2u(x))$, Pucci's maximal equation is defined as:

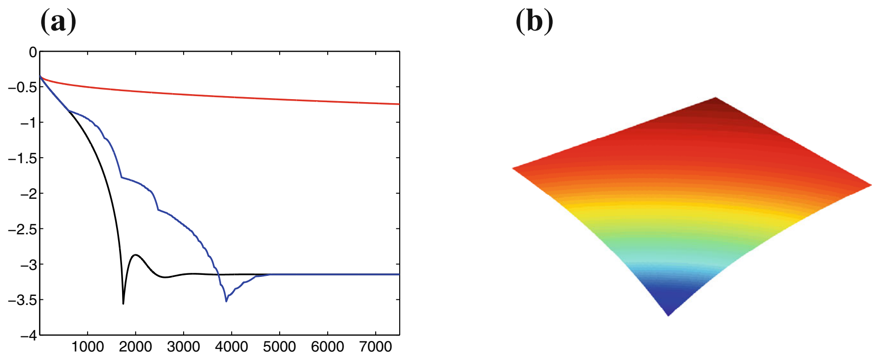


Fig. 5 Error and solution plots of the Pucci maximal equation on a 256-by-256 size grid. All three methods have the same initial conditions and time step parameter $\tau = \frac{h^2}{12}$. The first plot (*left*) shows the error (on a log scale) versus the number of iterations for the forward Euler method (*red*), Method 1 (*black*) and Method 2 (*blue*). Method 1 and Method 2 agree in this case as well. **a** Error, **b** solution (Color figure online)

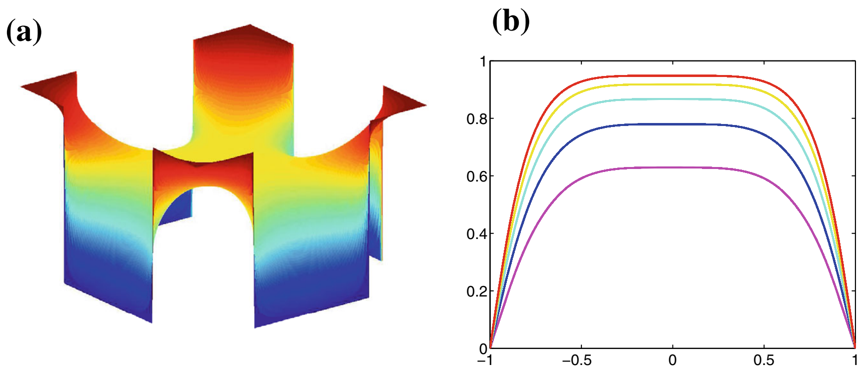


Fig. 6 Solution and slice plots of the Pucci maximal equation on a 256-by-256 size grid using Method 2. The stopping criteria is: $||u^{n+1} - u^n||_\infty \leq 10^{-7}$ and shows that the method will produce properly ordered solution, even away from the true steady state. **a** Solution, **b** solution slices (Color figure online)

$$\lambda_1 + \alpha\lambda_2 = 0,$$

with $\alpha > 1$, which is equivalent to:

$$\begin{cases} \alpha(\Delta u)^2 + (1-\alpha)^2 \det(D^2 u) = 0 \\ \Delta u \leq 0 \end{cases}$$

There are several ways of discretizing Pucci's equation which depend also on which form is taken above. For example, there are finite difference methods which use wide stencils [12] or finite element approaches using least squares optimization [21,45]. For this example, we use the wide stencils approach. In Fig. 5 and Table 3, we solve for the exact solution when $\alpha = 2$, i.e. $u(x, y) = -(x^2 + y^2)^{-1}$, on a 256-by-256. The initial data is the maximum of the boundary data. In Fig. 6, Method 2 is applied to the Pucci maximal equation with $\alpha = 2$ and with boundary data

Table 3 Convergence comparison between various methods using the example from Fig. 5

Grid size	M1	M2	FE
32	413	348	3272
64	849	841	13,457
128	1977	2037	75,768
256	4950	5028	238,751
512	10,144	11,785	–
Rate	1.18	1.27	2.11

$$g(x, y) = \begin{cases} 1, & \text{if } |x| > \frac{1}{2} \text{ and } |y| > \frac{1}{2} \\ 0, & \text{otherwise.} \end{cases}$$

Using the partial convergence criteria, Fig. 6 (right) shows slices of the solution for $\alpha = 2, 2.75, \ldots, 5$ (see also Table 4). The solutions have the correct ordering (via maximum principle), even though the solutions have not reached the true steady states.

6.5 Monge–Ampère Equation

The Monge–Ampère equation (in two dimensions) is defined as:

$$H(u) = \det(D^2 u) = \lambda_1 \lambda_2$$

and is elliptic over the space of convex functions. The Monge–Ampère equation arises in many mathematical and physical models, for example in geometry, cosmology, and fluid dynamics, and has received a growing amount of attention in recent years (see [2] and citations within). Numerical methods for the Monge–Ampère equation can be based on finite differences [12–16], finite elements using least squares and augmented Lagrangians [17–20,46] and finite elements using the vanishing moment [5,7–10]. For more detail on these numerical methods see [2].

For our first simulation, we use the monotone difference schemes found in [12,13]. We use the exact solution: $u(x, y) = e^{\frac{r^2}{2}}$ with the forcing term $f = e^{\frac{r^2}{2}}(1 + r^2)$, where $r^2 = x^2 + y^2$. The calculations are done on a 128-by-128 grid with $\tau = \frac{h^2}{75}$, see Fig. 7. The initial data is found by solving: $-\Delta u^0 = 0$ with the correct boundary data, thereby providing a supersolution. In this example, Method 1 and 2 have identical convergence rates.

For the remaining examples, we consider several singular solutions to the Monge–Ampère equation. The first singular solution has the following exact solution and forcing term:

$$u(x, y) = \frac{2\sqrt{2}}{3}((x-1)^2 + (y-1)^2)^{\frac{3}{4}} \tag{6.1}$$
$$f(x, y) = \frac{1}{\sqrt{(x-1)^2 + (y-1)^2}}.$$

The solution in Eq. (6.1) is continuous up to the boundary, but blows-up at boundary point (1, 1). In terms of the Monge–Ampère equation, we consider blow-up to refer to the derivatives of the function as opposed to the function values themselves. In Table 5, we use a multi-level approach to calculate the number of iteration needed to achieve convergence. For a given resolution, the solution is initialized by interpolating the values from the steady state

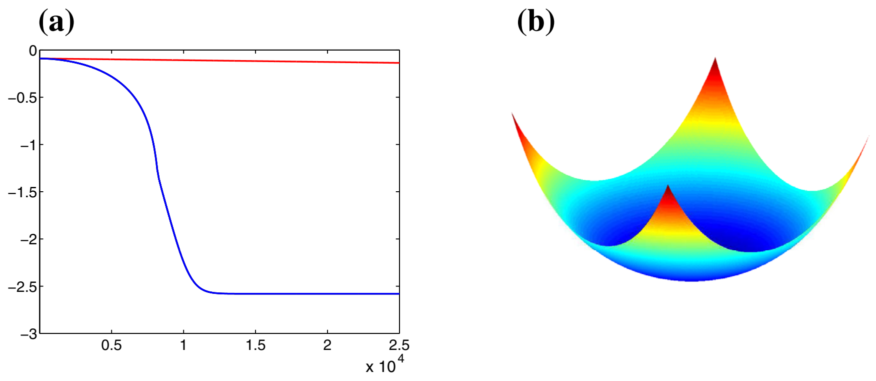


Fig. 7 Error and solution plots of the Monge–Ampère equation on a 128-by-128 size grid. All three methods have the same initial conditions and time step parameter $\tau = \frac{h^2}{75}$. The first plot (*left*) shows the error (on a log scale) versus the number of iterations for the forward Euler method (*red*), Method 1 (*black*) and Method 2 (*blue*). **a** Error, **b** solution (Color figure online)

Table 4 Dependence of the number of iteration needed on the parameter α (see in Fig. 6)

α	**Fixed** τ	**Varying** τ
2.00	2989	1888
2.75	2729	2022
3.50	2522	2110
4.25	2353	2169
5.00	2211	2211
Complexity	−0.33	0.17

Pucci's equation is solved on a 256-by-256 grid and the time-step is either fixed ($\tau = h^2/25$) or varies with alpha ($\tau = \alpha^{-1}h^2/5$). The iterations are terminated when the stopping criteria holds: $||u^{n+1} - u^n||_\infty \leq 10^{-7}$. The iteration complexity as a function of α is shown

Table 5 Convergence comparison between various methods using the example from Eq. (6.1)

Grid size	Error	M1	M2	FE
16	0.0250	564	601	2239
32	0.0160	585	651	2309
64	0.0110	976	1037	5759
128	0.0077	4626	4655	115,802
256	0.0066	6479	6541	219,158
Rate	0.48	1.0027	0.9726	1.8874

solution on a coarser resolution. The complexity is still linear (in terms of h) for our method; however, the total number of iterations are lower. The solution and the error versus iterations plots are shown in Fig. 8 and are calculated on a 128-by-128 grid with $\tau = \frac{h^2}{25}$.

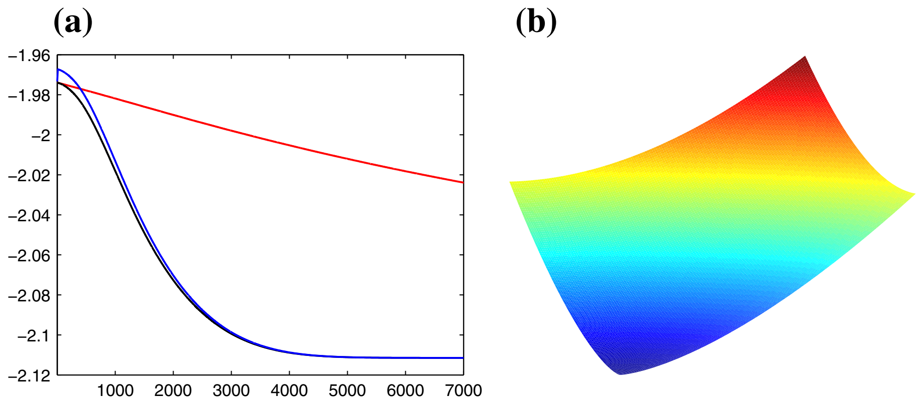


Fig. 8 Error and solution plots of the Monge–Ampère equation (Eq. (6.1)) on a 128-by-128 size grid. All three methods have the same initial conditions and time step parameter $\tau = \frac{h^2}{25}$. The first plot (*left*) shows the error (on a log scale) versus the number of iterations for the forward Euler method (*red*), Method 1 (*black*) and Method 2 (*blue*). **a** Error, **b** solution (Color figure online)

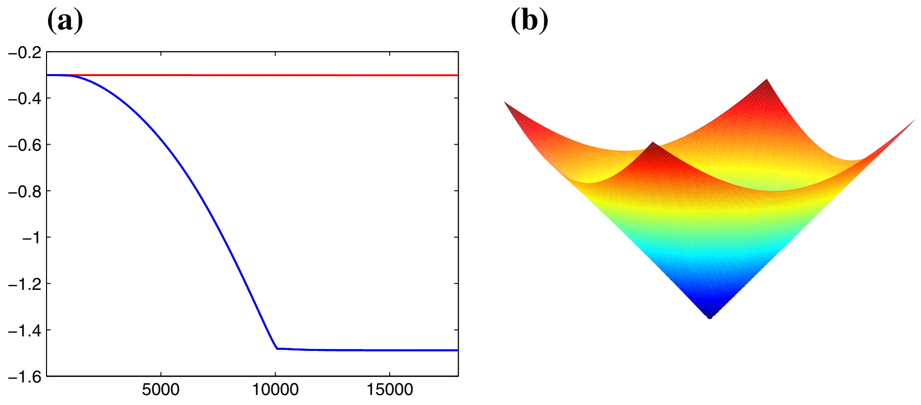


Fig. 9 Error and solution plots of the Monge–Ampère equation (Eq. (6.2)) on a 128-by-128 size grid. All three methods have the same initial conditions and time step parameter $\tau = \frac{h^2}{75}$. The first plot (*left*) shows the error (on a log scale) versus the number of iterations for the forward Euler method (*red*), Method 1 (*black*) and Method 2 (*blue*). The solution has a singularity at the point (0, 0). **a** Error, **b** solution (Color figure online)

The second example has the following exact solution and forcing term:

$$\begin{aligned} u(x, y) &= \sqrt{x^2 + y^2} \\ f(x, y) &= \delta_{(0,0)}(x, y), \end{aligned} \tag{6.2}$$

where δ_p is the Dirac delta function at the point p. Given the data in Eq. (6.2), the solution has singular second derivatives at (0, 0). The Dirac delta function is discretized as $\frac{1}{h^2}$ at (0, 0) and zero otherwise. The solution and the error versus iterations plots are shown in Fig. 9 and are calculated on a 128-by-128 grid with the stricter step size $\tau = h^4$. In Table 6, we see that using this time-step restriction results in a quadratic complexity in our method, but an even worse rate for the classical method.

Table 6 Convergence comparison between various methods using the example from Eq. (6.2). In this example, $\tau = h^4$, so we expect higher complexity as compared to other examples

Grid Size	Error	M1	M2	FE
16	0.22	159	159	1333
32	0.12	575	575	14101
64	0.0640	2571	2571	206,869
128	0.0320	14,503	14,503	–
256	0.0160	45,494	45,494	–
Rate	0.95	2.0978	2.0978	3.6389

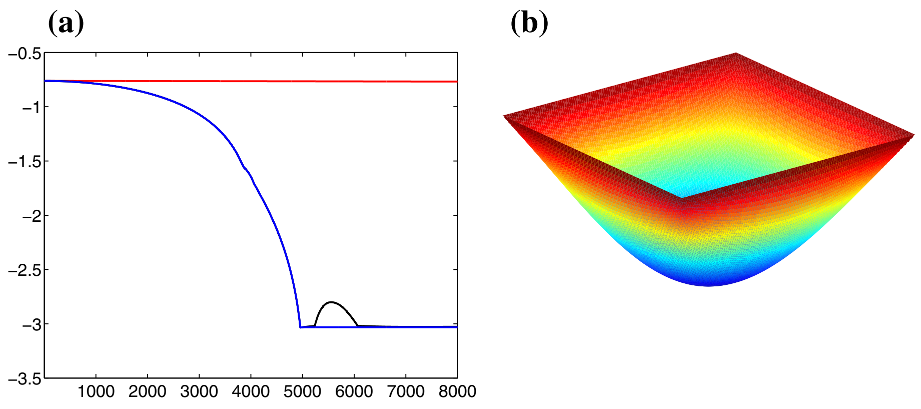


Fig. 10 Error and solution plots of the Monge–Ampère equation (Eq. (6.3)) on a 128-by-128 size grid. All three methods have the same initial conditions and time step parameter $\tau = \frac{h^2}{50}$. The first plot (*left*) shows the error (on a log scale) versus the number of iterations for the forward Euler method (*red*), Method 1 (*black*) and Method 2 (*blue*). **a** Error, **b** solution (Color figure online)

Table 7 Convergence comparison between various methods using the example from Eq. (6.3). In this example, $\tau = h^2/50$, so we expect higher complexity as compared to other examples

Grid size	M1	M2	FE
16	566	566	14,607
32	1199	1199	66,842
64	2481	2481	315,450
128	6287	6287	1,140,020
256	12,298	9948	–
Rate	1.1273	1.0662	2.1142

Using the constant forcing term:

$$f(x, y) = \frac{1}{16}, \tag{6.3}$$

and zero boundary data, we calculate the solution in Fig. 10 on a 128-by-128 grid with the step size $\tau = \frac{h^2}{50}$. Since the solution is zero on the boundary but the PDE is constant in the

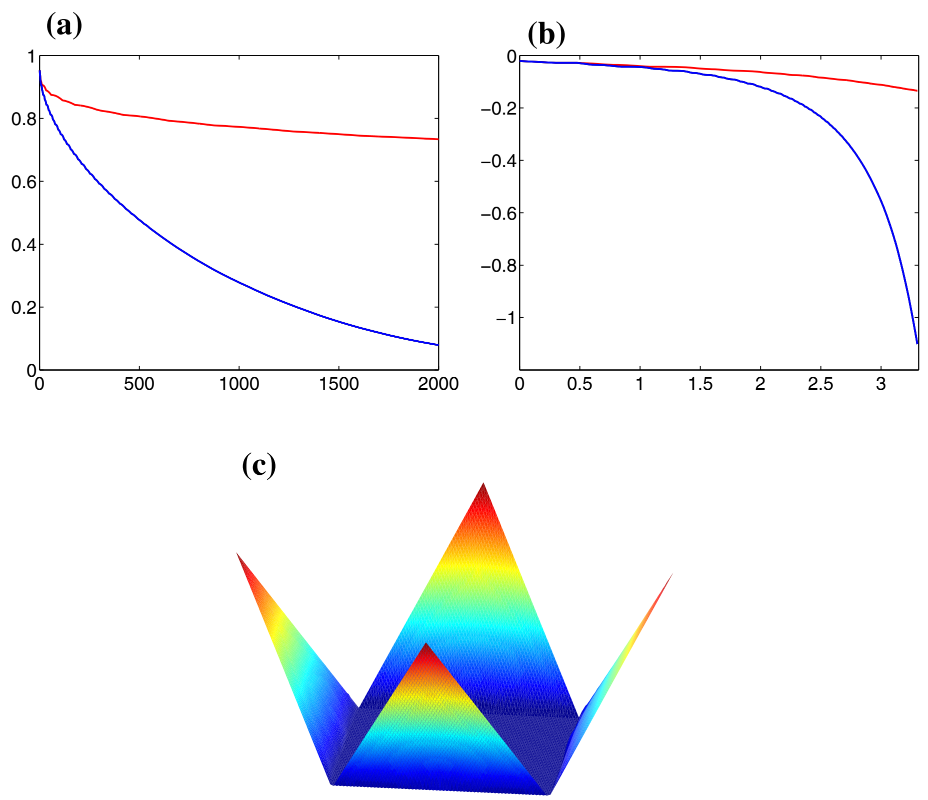


Fig. 11 Error versus iterations plots for Monge–Ampère equation (using the data from Eq. (6.4)) on 128-by-128 size grid. All three methods have the same initial conditions and time step parameter $\tau = \frac{h^4}{2}$. The first plot (*left*) shows the error (on a log scale) versus the number of iterations for the forward Euler method (*red*), Method 1 (*black*) and Method 2 (*blue*). The second plot (**b**) is the same as plot (**a**) but on a log–log scale. **a** Error log plot, **b** error log–log plot, **c** Solution (Color figure online)

interior, the second derivatives must blow-up as they approach the boundary. In Table 7, we calculate the convergence rate against a finer resolution solution.

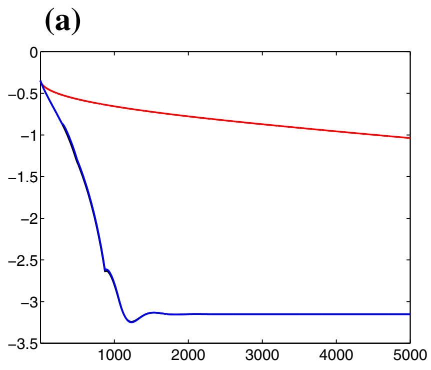
The last example has the following exact solution and forcing term:

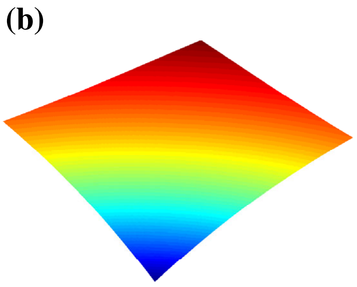
$$\begin{aligned} u(x, y) &= \max(|x| + |y|, 1) \\ f(x, y) &= 0. \end{aligned} \tag{6.4}$$

The solution has singular second derivatives on a domain of codimension one which is also not aligned exactly with the grid points. For this example, the error decreases monotonically, since the solution can be obtained exactly with the stencil. With initial data $u^0(x, y) = 1$ and time-step $\tau = h^4/2$, the solution and error versus iterations are shown in Fig. 11. The error is plotted one two different scales, showing significant acceleration over the standard method.

6.6 Systems of Fully Nonlinear Equations

For our final example we look at a system of fully nonlinear equations, namely the coupled Pucci system:





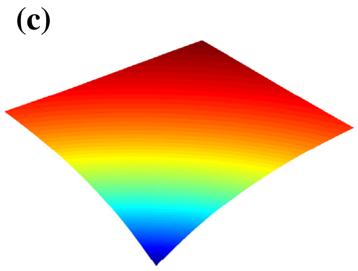


Fig. 12 Error and solution plots of a coupled system of Pucci maximal equations on a 128-by-128 size grid. All three methods have the same initial conditions and time step parameter $\tau = \frac{h^2}{12}$. The first plot (*left*) shows the error (on a log scale) versus the number of iterations for the forward Euler method (*red*), Method 1 (*black*) and Method 2 (*blue*). Method 1 and Method 2 agree in this case as well. The solutions (*bottom left and right*) are similar in shaped but have different curvatures rates. **a** Error, **b** solution 1, **c** solution 2 (Color figure online)

$$\begin{aligned}\alpha_1\lambda_1(D^2u) + \lambda_2(D^2u) + F_1(x, v) &= 0\\ \alpha_2\lambda_1(D^2v) + \lambda_2(D^2v) + F_2(x, u) &= 0\end{aligned}$$

We choose a quadratic coupling of zeroth order terms in the following way:

$$\begin{aligned}\frac{4}{3}\lambda_1(D^2u) + \lambda_2(D^2u) + v^2 &= f_1\\ 2\lambda_1(D^2v) + \lambda_2(D^2v) + u^2 &= f_2\end{aligned}$$

In Fig. 12, by choosing $(f_1, f_2)(x, y) = (-(x^2 + y^2)^{-\frac{2}{3}}, -(x^2 + y^2)^{-2})$ we have the exact solution $(u, v)(x, y) = (-(x^2 + y^2)^{-\frac{1}{3}}, -(x^2 + y^2)^{-1})$. The calculations are done on a 128-by-128 grid with the initial data taken to be the maximum of the solution. We see that Methods 1 and 2 converge at the same rate. Systems of nonlinear elliptic equation can be difficult to compute on finer grids due to the increase in complexity.

7 Conclusion

We presented two numerical methods for accelerating the convergence of iterative methods for nonlinear and/or degenerate elliptic PDE. We have shown that the first method is linearly stable and consistent to the solution of the PDE in which time is traversed at a quadratic rate. In the numerical experiments, we see that our method's complexity is the square root of the complexity of the standard iterative method. With an additional function evaluation, the second method is proven to be convergent in the viscosity sense. We have shown that experimentally, the methods converge faster than the standard fixed point methods with a nearly linear complexity.

Acknowledgments H. Schaeffer was supported by NSF 1303892 and University of California President's Postdoctoral Fellowship Program. T. Y. Hou was supported by NSF DMS 1318377, NSF DMS 0908546, and DOE DE FG02 06ER25727. The authors would like to thank the editor and reviewers for their helpful feedback.

Appendix

We provide a proof of Proposition 4.2 below. The arguments follows from a direct computation of the lower bound.

Proof We can expand α_n in terms of the sequences $\{\gamma_j\}_{j=1}^n$ as follows:

$$\begin{aligned}\alpha_n &= (1+\alpha_{n-1})\gamma_n = (1+(1+\alpha_{n-2})\gamma_{n-1})\gamma_n\\ &= (1+(1+(1+\alpha_{n-2})\gamma_{n-2})\gamma_{n-1})\gamma_n \dots\\ &= \sum_{j=1}^n \prod_{k=j}^n \gamma_k\end{aligned}$$

Next, define $\xi_k{:=}1-\gamma_k$, by the assumptions of Proposition 4.2, $1-\xi_k \geq 1-\frac{1}{k+1}$, so

$$\prod_{k=j}^n (1-\xi_k) \geq \prod_{k=j}^n (1-\frac{1}{k+1}) = \frac{j}{n+1}.$$

Therefore the sequence α_n can be bound below by:

$$\alpha_n = \sum_{j=1}^n \prod_{k=j}^n \gamma_k \geq \sum_{j=1}^n \frac{j}{n+1} = \frac{n}{2}.$$

Also, the partial sums of α_n are given by:

$$\sum_{j=1}^{n-1} \alpha_j \geq \sum_{j=1}^{n-1} \frac{j}{2} = \frac{n^2-n}{4}.$$

References

1. Falcone, M., Finzi Vita, S., Giorgi, T., Smits, R.G.: A semi-Lagrangian scheme for the game p-Laplacian via p-averaging. Appl. Numer. Math. **73**, 63–80 (2013)
2. Feng, X., Glowinski, R., Neilan, M.: Recent developments in numerical methods for fully nonlinear second order partial differential equations. SIAM Rev. **55**, 205–267 (2013)
3. Oberman, A.: A convergent difference scheme for the infinity Laplacian: construction of absolutely minimizing Lipschitz extensions. Math. Comput. **74**(251), 1217–1230 (2005)
4. Oberman, A.: Finite difference methods for the infinity Laplace and p-Laplace equations. J. Comput. Appl. Math. **254**, 65–80 (2013)

5. Feng, X., Neilan, M.: Mixed finite element methods for the fully nonlinear Monge–Ampère equation based on the vanishing moment method. SIAM J. Numer. Anal. **47**(2), 1226–1250 (2009)
6. Feng, X., Neilan, M.: Vanishing moment method and moment solutions for fully nonlinear second order partial differential equations. J. Sci. Comput. **38**(1), 74–98 (2009)
7. Feng, X., Neilan, M.: A modified characteristic finite element method for a fully nonlinear formulation of the semigeostrophic flow equations. SIAM J. Numer. Anal. **47**(4), 2952–2981 (2009)
8. Feng, X., Neilan, M.: Analysis of Galerkin methods for the fully nonlinear Monge–Ampère equation. J. Sci. Comput. **47**(3), 303–327 (2011)
9. Feng, X., Neilan, M.: Galerkin methods for the fully nonlinear Monge–Ampére equation. arXiv:0712.1240 (2007)
10. Neilan, M.: A nonconforming Morley finite element method for the fully nonlinear Monge–Ampère equation. Numer. Math. **115**(3), 371–394 (2010)
11. Lakkis, O., Pryer, T.: An adaptive finite element method for the infinity Laplacian. arXiv:1311.3930 (2013)
12. Oberman, A.M.: Wide stencil finite difference schemes for the elliptic Monge–Ampère equation and functions of the eigenvalues of the Hessian. Discrete Continuous Dyn. Syst. Ser. B **10**(1), 221–238 (2008)
13. Benamou, J.-D., Froese, B.D., Oberman, A.M.: Two numerical methods for the elliptic Monge–Ampère equation. ESAIM Math. Model. Numer. Anal. **44**(4), 737–758 (2010)
14. Froese, B.D., Oberman, A.M.: Convergent finite difference solvers for viscosity solutions of the elliptic Monge–Ampère equation in dimensions two and higher. SIAM J. Numer. Anal. **49**(4), 1692–1714 (2011)
15. Froese, B.D., Oberman, A.M.: Fast finite difference solvers for singular solutions of the elliptic Monge–Ampère equation. J. Comput. Phys. **230**(3), 818–834 (2011)
16. Froese, B.D., Oberman, A.M.: Convergent filtered schemes for the Monge–Ampère partial differential equation. SIAM J. Numer. Anal. **51**(1), 423–444 (2013)
17. Dean, E.J., Glowinski, R.: Numerical solution of the two-dimensional elliptic Monge–Ampère equation with Dirichlet boundary conditions: an augmented Lagrangian approach. Comptes Rendus Math. **336**(9), 779–784 (2003)
18. Dean, E.J., Glowinski, R.: Numerical solution of the two-dimensional elliptic Monge–Ampère equation with Dirichlet boundary conditions: a least-squares approach. Comptes Rendus Math. **339**(12), 887–892 (2004)
19. Dean, E.J., Glowinski, R.: An augmented Lagrangian approach to the numerical solution of the Dirichlet problem for the elliptic Monge–Ampére equation in two dimensions. Electron. Trans. Numer. Anal. **22**, 71–96 (2006)
20. Dean, E.J., Glowinski, R.: Numerical methods for fully nonlinear elliptic equations of the Monge–Ampère type. Comput. Methods Appl. Mech. Eng. **195**(13), 1344–1386 (2006)
21. Dean, E.J., Glowinski, R.: On the numerical solution of a two-dimensional Pucci's equation with Dirichlet boundary conditions: a least-squares approach. Comptes Rendus Math. **341**(6), 375–380 (2005)
22. Böhmer, K.: On finite element methods for fully nonlinear elliptic equations of second order. SIAM J. Numer. Anal. **46**(3), 1212–1249 (2008)
23. Barles, G., Souganidis, P.E.: Convergence of approximation schemes for fully nonlinear second order equations. Asympotot. Anal. **4**, 271–283 (1991)
24. Oberman, A.: Convergent difference schemes for degenerate elliptic and parabolic equations: Hamilton–Jacobi equations and free boundary problems. SIAM J. Numer. Anal. **44**(2), 879–895 (2006)
25. Caffarelli, L.A., Souganidis, P.E.: A rate of convergence for monotone finite difference approximations to fully nonlinear, uniformly elliptic PDEs. Commun. Pure Appl. Math. **61**, 1–17 (2008)
26. Loeper, G., Rapetti, F.: Numerical solution of the Monge–Ampè equation by a Newton's algorithm. Comptes Rendus Math. **340**(4), 319–324 (2005)
27. Awanou, G.: Standard finite elements for the numerical resolution of the elliptic Monge–Ampère equation: classical solutions. IMA J. Numer. Anal. http://imajna.oxfordjournals.org/content/early/2014/05/30/imanum.dru028 (2014)
28. Nesterov, Y.: A method of solving a convex programming problem with convergence rate O$(1/k^2)$. Sov. Math. Dokl. **27**(2), 372–376 (1983)
29. Beck, A., Taboulle, M.: A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM J. Imaging Sci. **2**(1), 183–202 (2009)
30. Hundsdorfer, W., Ruuth, S.J., Spiteri, R.J.: Monotonicity-preserving linear multistep methods. SIAM J. Numer. Anal. **41**(2), 605–623 (2003)
31. Gottlieb, S., Shu, C.-W., Tadmor, E.: Strong stability-preserving high-order time discretization methods. SIAM Rev. **43**(1), 89–112 (2001)
32. Gottlieb, S.: On high order strong stability preserving Runge–Kutta and multi step time discretizations. J. Sci. Comput. **25**(1), 105–128 (2005)

33. Ruuth, S.: Global optimization of explicit strong-stability-preserving Runge–Kutta methods. Math. Comput. **75**(253), 183–207 (2006)
34. Bresten, C., Gottlieb, S., Grant, Z., Higgs, D., Ketcheson, D. I., Németh, A.: Strong stability preserving multistep Runge–Kutta methods. arXiv:1307.8058 (2013)
35. Kuo, H.J., Trudinger, N.S.: Positive difference operators on general meshes. Duke Math. J. **83**(2), 415–433 (1996)
36. Isaacs, R.: Differential Games. Wiley, New York (1965)
37. Cabré, X., Caffarelli, L.A.: Interior $C^{2,\alpha}$ regularity theory for a class of non convex fully nonlinear elliptic equation. J. Math. Pures Appl. **82**(5), 573–612 (2003)
38. Krylov, N.V.: On the rate of convergence of finite-difference approximations for elliptic Isaacs' equation in smooth domains. arXiv:1402.0252 (2014)
39. Barron, E., Evans, L.C., Jensen, R.: The infinity Laplacian. Aronsson's equation and their generalizations. Trans. Am. Math. Soc. **360**(1), 77–101 (2008)
40. Evans, L.C., Yu, Y.: Various properties of solutions of the infinity-Laplacian equation. Commun. Partial Differ. Equ. **30**(9), 1401–1428 (2005)
41. Crandall, M.G., Evans, L.C., Gariepy, R.F.: Optimal Lipschitz extensions and the infinity Laplacian. Calc. Var. Partial Differ. Equ. **13**(2), 123–139 (2001)
42. Evans, L.C., Smart, C.K.: Everywhere differentiability of infinity harmonic functions. Calc. Var. Partial Differ. Equ. **42**(1–2), 289–299 (2011)
43. Aronsson, G.: On the partial differential equation $u_x^2 u_{xx} + 2u_x u_y u_{xy} + u_y^2 u_{yy} = 0$. Ark. Mat. **7**(5), 395–425 (1968)
44. Aronsson, G.: On certain singular solutions of the partial differential equation $u_x^2 u_{xx} + 2u_x u_y u_{xy} + u_y^2 u_{yy} = 0$. Manuscr. Math. **47**, 133–151 (1984)
45. Caffarelli, L.A., Glowinski, R.: Numerical solution of the Dirichlet problem for a Pucci equation in dimension two. Application to homogenization. J. Numer. Math. **16**(3), 185–216 (2008)
46. Caboussat, A., Glowinski, R., Sorensen, D.C.: A least-squares method for the numerical solution of the Dirichlet problem for the elliptic Monge–Ampère equation in dimension two. ESAIM Control Optim. Calc. Var. **19**(03), 780–810 (2013)