ANZIAM J. 61(2019), 23–46
doi:10.1017/S1446181118000391

APPLICATION OF PROJECTION ALGORITHMS TO DIFFERENTIAL EQUATIONS: BOUNDARY VALUE PROBLEMS

BISHNU P. LAMICHHANE[1], SCOTT B. LINDSTROM[✉1] and BRAILEY SIMS[1]

(Received 31 May, 2017; accepted 27 August, 2018; first published online 18 February 2019)

Abstract

The Douglas–Rachford method has been employed successfully to solve many kinds of nonconvex feasibility problems. In particular, recent research has shown surprising stability for the method when it is applied to finding the intersections of hypersurfaces. Motivated by these discoveries, we reformulate a second order boundary value problem (BVP) as a feasibility problem where the sets are hypersurfaces. We show that such a problem may always be reformulated as a feasibility problem on no more than three sets and is well suited to parallelization. We explore the stability of the method by applying it to several BVPs, including cases where the traditional Newton's method fails.

2010 *Mathematics subject classification*: primary 34B15; secondary 47H10.

Keywords and phrases: boundary value problem, Douglas–Rachford method, Newton's method, hypersurface.

1. Introduction

We explore a particular approach to obtaining approximate numerical solutions to (second order, nonlinear) boundary value problems (BVPs) on $[a, b] \subseteq \mathbb{R}$. We use finite difference approximations to replace the continuous problem by a discrete one involving a finite system of N nonlinear equations in N variables (that is, the approximate solution values at each of the N partition points). The classical approach to solving such a system of equations is to use Newton's method. We explore some alternative projection-based iterative methods.

The solution set for each of the N equations is a hypersurface S_k in N-dimensional Euclidean space $\mathbb{R}^N$. An approximate solution to the BVP then corresponds to a point

[1]Priority Research Centre for Computer-Assisted Research Mathematics and its Applications (CARMA), University of Newcastle, New South Wales, Australia;
e-mail: bishnu.lamichhane@newcastle.edu.au, scott.lindstrom@uon.edu.au, brailey.sims@newcastle.edu.au.

in the intersection of these N hypersurfaces. This is a *feasibility* problem of the form

$$\text{find} \quad x \in \bigcap_{k=1}^{N} S_k.$$

One approach to solving such feasibility problems is to use an iterated process.

We consider the method of alternating projections (AP) and the Douglas–Rachford method (DR) in particular. We explain how the methods are well suited to parallelization. We then use the methods to solve the associated feasibility problems for several BVPs and compare the results with those given by the classical Newton's method.

1.1. Objectives Our intent is not to compare the speeds of our projection-based methods with that of Newton's method, which is much faster, neither is it our goal to provide a full comparison of their respective robustness. The main contributions are as follows.

(1) We introduce the reformulation of ordinary differential equations (ODEs) as hypersurface feasibility problems for solving with iterated projection methods.
(2) We show how they are particularly amenable to parallelization.
(3) For boundary value ODEs, we show how to reformulate the N-hypersurface feasibility problem as a three-set feasibility problem.
(4) We analyse the behaviour for both AP and DR experimentally on hypersurface problems for varying N, which for boundary value ODEs corresponds to partition fineness. We catalogue the characteristics of oscillation so frequently observed for DR in particular.
(5) We provide a characterization of how it might be employed to real world systems of equations in cases where Newton's method does not succeed.

This work extends to N sets via Pierra's method [24] (also known as the *divide and concur* method [17]), and the two-set investigation started by Borwein and Sims [9], who analysed DR for the hypersurfaces choices of an $(n-1)$-sphere and a line. In this simpler setting, global convergence was shown by Aragón Artacho and Borwein [1] under an assumption later relaxed by Benoist [7], who demonstrated convergence by means of a Lyapunov function. The analysis has already been extended in $\mathbb{R}^2$ by Borwein et al. [8], who considered the generalization of circles to ellipses and p-spheres. Later, Lindstrom et al. considered plane curves more generally [21]. Inspired by Benoist's work [7], Dao and Tam [14] have since provided a beautiful illumination of the method for curves in $\mathbb{R}^2$ by means of Lyapunov functions. Phan [23] and later Dao and Phan [13] have since provided more general convergence results under regularity (transversality) conditions.

While this article is an important extension of the analysis of projection methods (and DR in particular) for nonconvex hypersurface feasibility problems, its approach is comparable to other experimental works which analyse proximal point algorithms in

https://doi.org/10.1017/S1446181118000391 Published online by Cambridge University Press

the absence of convexity by cataloguing the performance of the method for a selection of examples. These include the recent work of Aragón Artacho et al. [3] applying DR to solve matrix completion problems and Sudoku puzzles [2, 10], the work of Aragón Artacho and Campoy [4] with graph colouring problems and the seminal work of Elser et al. [16].

We have listed here only a small selection of the nonconvex Douglas–Rachford genre. The history is vast, and we have not even touched on its roots in convex optimization and connections with the celebrated ADMM (alternating direction method of multipliers) through duality. For a more thorough treatment, we refer the reader to a recent survey of Lindstrom and Sims [20].

1.2. Outline The outline of this paper is as follows. In Section 2 we introduce nonlinear boundary value problems. In Section 3 we introduce the two-set projection algorithms and their extension to N sets. In Section 4 we discuss methods of projecting onto individual hypersurfaces and in Section 5 we describe the full procedure, discuss its amenability to parallelization and show a natural reformulation which reduces the N-set problem to a three-set problem. We provide our experimental results and conclusion in Section 6.

2. Nonlinear boundary value problems

We investigate the use of projection algorithms to obtain numerical solutions to nonlinear boundary value problems. Here and throughout:

(i) $y : [a, b] \subset \mathbb{R} \to \mathbb{R}$ with $a < b$ is an "unknown" function for which we seek a numerical solution;
(ii) y' and y'' are, respectively, the first and second derivatives of y;
(iii) $\alpha = y(a) \in \mathbb{R}$ and $\beta = y(b) \in \mathbb{R}$ are given boundary values.

A complete statement of the problem is

$$\begin{aligned} &\text{given a function } f : \mathbb{R}^3 \to \mathbb{R}, \text{ find } y \text{ such that} \\ &y'' = f(x, y, y') \quad \text{for } x \in (a, b) \subset \mathbb{R} \text{ with } y(a) = \alpha \text{ and } y(b) = \beta. \end{aligned} \tag{2.1}$$

Remark 2.1 (Solutions may not be unique). In general, even when a solution to problem (2.1) exists, it may not be unique. However, (2.1) will have a unique continuous solution over the interval $[a, b]$ if the right-hand-side function f satisfies the following conditions:

(1) f and the partial derivatives of f with respect to y and y' are continuous on

$$D = \{(x, y, y') \,|\, a \le x \le b, -\infty < y < \infty, -\infty < y' < \infty\};$$

(2) $(\partial f/\partial y)(x, y, y') > 0$ on D; and

https://doi.org/10.1017/S1446181118000391 Published online by Cambridge University Press

(3) there exists a constant M such that

$$\left|\frac{\partial f}{\partial y'}(x, y, y')\right| \leq M \quad \text{on } D \tag{2.2}$$

(see, for example, the book by Burden et al. [12, Theorem 11.1]). Because we seek to present the wide variety of behaviours exhibited by our algorithms, we present both examples which may and may not satisfy these criteria.

We use a finite difference method to approximate the solution of the given BVP. This results in a system of nonlinear equations to which we apply our projection algorithm to compute an approximate numerical solution.

To this end, consider a partition of the interval $[a, b]$ into N equal subintervals using the set of points $x_i = a + ih$ for $i = 1, \ldots, N$ with $x_0 = a$ and $x_{N+1} = b$, so that

$$h = \frac{b-a}{N+1}.$$

We introduce the centred-difference approximations for $i = 1, \ldots, N$,

$$y'(x_i) \approx \frac{y(x_{i+1}) - y(x_{i-1})}{2h}$$

and

$$y''(x_i) \approx \frac{y(x_{i+1}) - 2y(x_i) + y(x_{i-1})}{h^2}.$$

When the exact solution y is four times continuously differentiable these estimate the first and second derivatives at x_i with errors of $h^2 y^{(3)}(\eta_i)/6$ and $h^2 y^{(4)}(\xi_i)/12$, respectively, where η_i and ξ_i lie in the interval (x_{i-1}, x_{i+1}).

Ignoring such truncation error terms, we replace the first and second derivatives of y by their centred-difference approximations in (2.1) to obtain for $i = 1, 2, 3, \ldots, N$ the approximate relationships

$$\frac{y(x_{i+1}) - 2y(x_i) + y(x_{i-1})}{h^2} \approx f\left(x_i, y(x_i), \frac{y(x_{i+1}) - y(x_{i-1})}{2h}\right).$$

This leads us to take $y(x_i) \approx \omega_i$ as an approximate numerical solution to (2.1), where the ω_i satisfy the system of generally nonlinear equations

$$\omega_0 = \alpha, \quad \omega_{N+1} = \beta \quad \text{and}$$
$$\frac{\omega_{i+1} - 2\omega_i + \omega_{i-1}}{h^2} - f\left(x_i, \omega_i, \frac{\omega_{i+1} - \omega_{i-1}}{2h}\right) = 0 \quad \text{for } i = 1, 2, 3, \ldots, N.$$

If $h < 2/M$, where M is as defined in (2.2), and the other conditions of Remark 2.1 are satisfied, then this nonlinear system of equations has a unique solution [18, p. 86]. While many of our examples do not satisfy the conditions of Remark 2.1, uniqueness

https://doi.org/10.1017/S1446181118000391 Published online by Cambridge University Press

implies that we can easily measure the accuracy of a numerical approach, whereas if we have nonuniqueness it is much harder. For $i = 1, 2, \ldots, N$, let

$$\varphi_i(\omega) = \frac{\omega_{i+1} - 2\omega_i + \omega_{i-1}}{h^2} - f\left(x_i, \omega_i, \frac{\omega_{i+1} - \omega_{i-1}}{2h}\right) \quad \text{and} \tag{2.3}$$

$$\Omega_i = \{\omega = (\omega_1, \ldots, \omega_N) \mid \omega \text{ satisfies } \varphi_i(\omega) = 0\}. \tag{2.4}$$

Then we can compute our approximate numerical solution to the BVP (2.1) by solving the feasibility problem: find $\omega \in \bigcap_{i=1}^{N} \Omega_i$. An approximate numerical solution to (2.1) is then given by $y(x_i) = \omega_i$. For the task, we employ both the method of alternating projections and a parallelized version of DR as outlined below.

Remark 2.2. The astute reader will note that more complicated boundary conditions may be handled by appropriately modifying either or both of the equations $\omega_0 = \alpha$, $\omega_{N+1} = \beta$ though this could potentially lead to an enlarged problem of $N + 2$ equations in $N + 2$ unknowns. For example, the mixed condition $y(a) + \eta y'(a) = \alpha$ could translate to $\varphi_0(\omega) = \omega_0 + \eta(\omega_1 - \omega_0)/h = \alpha$.

3. Preliminaries on projection methods

The DR and AP are frequently used to find a feasible point (a point in the intersection) of two closed constraint sets A and B in a Hilbert space, in our setting: N-dimensional Euclidean space, $\mathbb{R}^N$.

The projection onto a subset C of $\mathbb{R}^N$ is defined for all $x \in \mathbb{R}^N$ by

$$\mathbb{P}_C(x) = \left\{z \in C \mid \|x - z\| = \inf_{z' \in C} \|x - z'\|\right\}.$$

> Note that $\mathbb{P}_C$ is a set-valued map where values may be empty or contain more than one point. In our case of interest, where C is a closed hypersurface, $\mathbb{P}_C$ has nonempty values and, in order to simplify both notation and implementation, we will work with a selector for $\mathbb{P}_C$, that is, a map $P_C : \mathbb{R}^N \to C : x \mapsto P_C(x) \in \mathbb{P}_C(x)$.

When C is nonempty, closed and convex, the projection operator P_C is uniquely determined and *firmly nonexpansive*, that is, for all $x, y \in \mathbb{R}^N$,

$$\|P_C x - P_C y\|^2 + \|(I - P_C)x - (I - P_C)y\|^2 \leq \|x - y\|^2.$$

See, for example, the book by Bauschke and Combettes [5, Ch. 4]. When C is a closed subspace, it is also a linear operator [5, Corollary 3.22].

The reflection mapping through the set C is then defined by

$$R_C = 2P_C - I,$$

where I is the identity map.

https://doi.org/10.1017/S1446181118000391 Published online by Cambridge University Press

Definition 3.1 (Method of alternating projections). For two closed sets A and B and an initial point $x_0 \in H$, the method of alternating projections (AP) generates a sequence $(x_n)_{n=1}^{\infty}$ as follows:

$$x_{n+1} \in T'_{A,B}(x_n) \quad \text{where } T'_{A,B} = P_B P_A.$$

DR was introduced half a century ago in connection with nonlinear heat flow problems [15].

Definition 3.2 (Douglas–Rachford method). For two closed sets A and B and an initial point $x_0 \in H$, the Douglas–Rachford method (DR) generates a sequence $(x_n)_{n=1}^{\infty}$ as follows:

$$x_{n+1} \in T_{A,B}(x_n) \quad \text{where } T_{A,B} = \tfrac{1}{2}(I + R_B R_A).$$

Definition 3.3 (Fixed point set). The fixed point set for an operator T is $\operatorname{Fix} T = \{x \in H \mid x \in Tx\}$.

The following theorem of Bauschke et al. [6] relaxes, in the context of convex feasibility, previous convergence conditions established in a somewhat different context by Lions and Mercier [22] (see also the article by Svaiter [26]).

Theorem 3.4 [6, Fact 5.9]. *Suppose that $A, B \subseteq H$ are closed and convex with nonempty intersection. Given $x_0 \in H$, the sequence of iterates $\{x_n\}$ defined by $x_{n+1} = T_{A,B}x_n$ converges weakly to an $x \in \operatorname{Fix} T_{A,B}$ with $P_A x \in A \cap B$.*

Of course, in our case, where the space is finite dimensional, weak convergence ensures convergence in norm.

Not withstanding the absence of a satisfactory theoretical justification, the DR iteration scheme has been used to successfully solve a wide variety of practical problems in which one or both of the constraints are nonconvex.

In an effort to develop the beginnings of a theoretical basis for employment in the nonconvex setting, Borwein and Sims [9] explored a feasibility problem on two particular hypersurfaces in $\mathbb{R}^n$: a line and the $(n-1)$-sphere. Among other results, they established local convergence near each of the (possibly two) feasible points. More extensive regions of convergence were determined by Borwein and Aragón Artacho [1]. The definitive answer, as conjectured by Borwein and Sims [9], was subsequently given by Benoist [7], who established convergence to the nearest feasible point except for starting points lying on a singular set: the hyperplane of symmetry.

Borwein et al. [8] showed that local convergence still holds for a line and a smooth hypersurface in $\mathbb{R}^N$ not intersecting asymptotically, although the basis of convergence may be quite sensitive to small perturbations of the sets. Additionally, Lindstrom et al. [21] extended local convergence to isolated points of intersection for two smooth hypersurfaces in $\mathbb{R}^N$. The authors of [19] showed local convergence for Neumann's method of alternating projections for sets under regularity conditions. Phan [23], and later Phan and Dao [13], showed local convergence with *R-linear* convergence rate for the strongly regular system $\{A, B\}$ of superregular sets A, B. For more details on the history, we again refer the reader to [20].

https://doi.org/10.1017/S1446181118000391 Published online by Cambridge University Press

3.1. Extension of DR to multiple sets We can apply this method to a consistent feasibility problem with N sets $\Omega_1 \cdots \Omega_N \subset \mathbb{R}^N$ to find $x \in \bigcap_{k=1}^N \Omega_k \neq \emptyset$. We do so by working in the product space $\mathbb{R}^{N\times N}$ as follows. Let

$$A = \Omega_1 \times \cdots \times \Omega_N \quad \text{and} \quad B = \{(x_1, \ldots, x_N) \in \mathbb{R}^{N\times N} \mid x_1 = x_2 = \cdots = x_N\} \tag{3.1}$$

and apply DR to the two sets A and B. The product space projections for $x = (x_1, \ldots, x_N) \in \mathbb{R}^{N\times N}$ are

$$P_A(x_1, \ldots, x_N) = (P_{\Omega_1}(x_1), \ldots, P_{\Omega_N}(x_N)),$$
$$P_B(x_1, \ldots, x_N) = \left(\frac{1}{N}\sum_{k=1}^N x_k, \ldots, \frac{1}{N}\sum_{k=1}^N x_k\right).$$

This is sometimes called the "divide and concur" method (see, for example, the articles by Gravel and Elser [17] and Pierra [24]). This method is particularly well suited to parallelization. An alternative would be to use the cyclic DR algorithm introduced in [11].

We consider, in particular, the case where the Ω_i are as in (2.4). Where $\omega_0 = \alpha$ and $\omega_{N+1} = \beta$ are fixed, the *feasibility problem* is reduced to finding a point in the intersection of a family of N hypersurfaces $\Omega_1, \ldots, \Omega_N$ in $\mathbb{R}^N$.

4. Computations for nearest-point projection onto a surface P_{Ω_k}

For a hypersurface Ω_k in $X = \mathbb{R}^N$ implicitly defined by

$$x \in \Omega_k \iff \phi_k(x) = 0,$$

the nearest-point projection $x = P_{\Omega_k}(u)$ of $u \in \mathbb{R}^N$ onto Ω_k solves

$$\begin{aligned} &\text{minimize} \quad F(x) = \tfrac{1}{2}\|u - x\|^2 \\ &\text{subject to} \quad \phi_k(x) \;=\; 0. \end{aligned} \tag{4.1}$$

So, provided $u \notin \Omega_k$ and assuming sufficient differentiability, we know by the theory of Lagrange multipliers [25] that there exists $\lambda_0 \neq 0$ for which $(P_{\Omega_k}(u), \lambda_0)$ is a critical point of the Lagrangian $\mathcal{L}(x, \lambda) = F(x) - \lambda\phi_k(x)$. That is, $\lambda = \lambda_0$ and $x = P_{\Omega_k}(u)$ is a solution of

$$u - x + \lambda\nabla_x\phi_k(x) = 0 \quad \text{and} \quad \phi_k(x) = 0. \tag{4.2}$$

Again assuming sufficient differentiability to ensure that the $(N + 1) \times (N + 1)$ Jacobian

$$J(x, \lambda)_{l,j} = \begin{pmatrix} \lambda\left[\dfrac{\partial^2}{\partial x_l \partial x_j}\phi\right] - I & [\nabla_x\phi_l(x)]^{\mathrm{T}} \\ \nabla\phi_x(x) & 0 \end{pmatrix}$$

is well defined, the nonlinear system (4.2) could be solved using Newton's method. However, this requires solving, at each iteration, the system of $N + 1$ equations given by $J(x, \lambda)v = b$. The quasi-Newton method requires solving a similar linear system.

https://doi.org/10.1017/S1446181118000391 Published online by Cambridge University Press

Alternatively, we might seek to locate a point (x_0, λ_0) where the scalar function

$$G(x, \lambda) = \tfrac{1}{2}\{\|u - x + \lambda \nabla_x \phi(x)\|^2 + \phi(x)^2\}$$

has a minimum zero; $P_{\mathcal{S}}(u) = x_0$ is then the desired solution. For this, we might use the method of gradient (steepest) descent with a line search implemented at each iteration. This obviates the need to invert $J(x, \lambda)$, but, depending on the method employed for the line search, may involve performing several iterations of Newton's method on a one-variable function at each step. The main difficulty here is choosing a suitable starting point; $(u, 0)$ is one choice.

A simple code for computing the hypersurface projections for $\Omega_1, \ldots, \Omega_N$ may be seen in Algorithm 1.

5. The procedure

To move from a given iterate to a successive iterate, one must compute the approximate projections $P_{\Omega_k}(u), k = 1, \ldots, N$. One may use an appropriate iterative numerical method to solve the subproblem (4.1), continuing the method until successive iterates differ by less than some pre-prescribed tolerance τ.

The choice of numerical method is between needing more steps but less computational complexity (without the Jacobian) versus needing fewer steps with each entailing greater computational complexity (with the Jacobian). For the sake of simplicity, we used the Jacobian for all of our experiments and computed until the change from one step to the next was less than 10^{-12}.

Even though $\mathcal{P}_{\Omega_k}(x_{m,k})$, as a (possibly rough) numerical approximation to P_{Ω_k}, may not lie exactly on the surface Ω_k, we naturally use $\mathcal{R}(x_{m,k}) = (2\mathcal{P}_{\Omega_k} - I)(x_{m,k})$ in place of the reflection of $x_{m,k}$ in $\Omega_{m,k}$ when computing the $(m + 1)$th iterate of the Douglas–Rachford algorithm, so that

$$x_{m+1,k} = \frac{1}{N}\left(\sum_{j=1}^{N} \mathcal{R}_{\Omega_j}(x_{m,k})\right) - \frac{1}{2}\mathcal{R}_{\Omega_k}(x_{m,k}) + \frac{1}{2}x_{m,k}.$$

Remark 5.1. One might consider using a tolerance τ_m that reduces as the number of iterations increases and hopefully move nearer to a solution. Otherwise, it is unlikely that the accuracy of the solution found would exceed the preselected tolerance, τ. One could use $\tau_m = \alpha \operatorname{diam}\{x_{m,k} \mid k = 1, 2, \ldots, N\}$, where $\alpha \in (0, 1)$ and

$$\operatorname{diam}(S) = \max_{s_i, s_j \in S} \|s_i - s_j\|.$$

While theory does not guarantee convergence with either method of computing projections, experimentation has shown that for some of the problems DR may be relatively insensitive to small changes in how projections are computed [21]. This is why it makes sense to consider adapting the tolerance over successive iterates.

https://doi.org/10.1017/S1446181118000391 Published online by Cambridge University Press

Algorithm 1: Compute $\Omega_1, \dots, \Omega_N \subset \mathbb{R}^N$ and projections onto them.

procedure *Generate hypersurfaces*

Data: Receives as input a function f, which defines the differential equation, boundary points a, b with corresponding solution values α, β, which define the boundary conditions, and a number N of partition points.

Result: Returns $\phi = (\phi_1, \dots, \phi_N)$, where $\Omega_k = \{z \mid \phi_k(z) = 0\}$ is the kth hypersurface for the feasibility problem.

set $h := (b - a)/(N + 1)$;

for $k \in \{1, \dots, N\}$ **do**

  if $k = 1$ **then**

    set $\phi_k := x \mapsto 2x_k - x_{k+1} + h^2 f(a + kh, x_k, (x_{k+1} - \alpha)/2h) - \alpha$;

  else if $k = N$ **then**

    set $\phi_k := x \mapsto 2x_k - \beta + h^2 f(a + kh, x_k, (\beta - x_{k-1})/2h) - x_{k-1}$;

  else

    set $\phi_k := x \mapsto 2x_k - x_{k+1} + h^2 f(a + kh, x_k, (x_{k+1} - x_{k-1})/2h) - x_{k-1}$;

store ϕ;

procedure *Compute Lagrangian problems*

Data: receives as input the N-tuple ϕ

Result: Stores $\varphi = (\varphi_1, \dots, \varphi_N)$, where $\varphi_k = \{\varphi_{k,1}, \varphi_{k,2}, \varphi_{k,3}\}$ are three of the four functions from the Lagrangian system for computing a projection onto Ω_k (the fourth function is ϕ_k).

for $k \in \{1, \dots, N\}$ **do**

  set $\varphi_{k,2} := (v, u, \lambda) \mapsto 2u_k - 2v_k - \lambda \partial_k \phi_k(u)$;

  if $k = 1$ **then**

    set $\varphi_{k,1} := (v, u, \lambda) \mapsto 0$;

  else if $k = N$ **then**

    $\varphi_{k,3} := (v, u, \lambda) \mapsto 0$;

  else

    set $\varphi_{k,1} := (v, u, \lambda) \mapsto 2u_{k-1} - 2v_{k-1} - \lambda \partial_{k-1} \phi_k(u)$;

    set $\varphi_{k,3} := (v, u, \lambda) \mapsto 2u_{k+1} - 2v_{k+1} - \lambda \partial_{k+1} \phi_k(u)$;

store φ;

procedure *Projection for* Ω_k

Data: receives as input a value $k \in \{1, \dots, N\}$ and a value $x \in \mathbb{R}^N$.

Result: Returns a point $u \in P_{\Omega_k}(x)$.

Numerically solve the system

$\{\phi_k(u) = 0, \varphi_{k,1}(x, u, \lambda) = 0, \varphi_{k,2}(x, u, \lambda) = 0, \varphi_{k,3}(x, u, \lambda) = 0\}$ for u;

one may use, for example, Algorithm 2 or Algorithm 3.

return u;

https://doi.org/10.1017/S1446181118000391 Published online by Cambridge University Press

Algorithm 2: Projects a point x onto a set Ω_k with Newton's method

procedure *Projection for* Ω_k

Data: receives as input a value $k \in \{1, \dots, N\}$, a value $x \in \mathbb{R}^N$ and a threshold Γ

Result: Returns a point $u \in P_{\Omega_k}(x)$.

Set $G := (v, \lambda) \mapsto (\phi_k(v), \varphi_{k,1}(x, v, \lambda), \varphi_{k,2}(x, v, \lambda), \varphi_{k,3}(x, v, \lambda))$.

Set $J := (v, \lambda) \to J(v, \lambda)$, where $J(v, \lambda)$ is the Jacobian of G evaluated at (v, λ).

Set Newt $:= (v, \lambda) \mapsto (v, \lambda) - \Big((\text{MatrixInverse}(J(v, \lambda))).G(v, \lambda)\Big)$, where the dot denotes multiplication of a vector by a matrix.

Set $\eta_{\text{old}} = (x, 1)$.

Set $\eta_{\text{new}} = \text{Newt}(x, 1)$.

while $\|\eta_{old} - \eta_{new}\| > \Gamma$ **do**

Set $\eta_{\text{old}} := \eta_{\text{new}}$.

Set $\eta_{\text{new}} := \text{Newt}(\eta_{\text{old}})$.

Set $u := \eta_{\text{new}}$.

Return $(u_1, \dots, u_N)$, the first N components of u. Note that the $(N + 1)$th component was merely the final Lagrange multiplier.

Algorithm 3: Projects a point x onto a set Ω_k with steepest descent method

procedure *Projection for* Ω_k

Data: receives as input a value $k \in \{1, \dots, N\}$, a value $x \in \mathbb{R}^N$, a step size modifier γ and a threshold Γ

Result: Returns a point $u \in P_{\Omega_k}(x)$.

Set $G := (v, \lambda) \mapsto (\phi_k(v))^2 + (\varphi_{k,1}(x, v, \lambda))^2 + (\varphi_{k,2}(x, v, \lambda))^2 + (\varphi_{k,3}(x, v, \lambda))^2$.

Set $G'(v, \lambda)$ as the gradient of G evaluated at (v, λ).

Set Descent $:= (v, \lambda) \mapsto v - \gamma G'(v, \lambda)$.

Set $\eta_{\text{old}} = (x, 1)$.

Set $\eta_{\text{new}} = \text{Descent}(x, 1)$.

while $\|\eta_{old} - \eta_{new}\| > \Gamma$ **do**

Set $\eta_{\text{old}} := \eta_{\text{new}}$.

Set $\eta_{\text{new}} := \text{Descent}(\eta_{\text{old}})$.

Set $u := \eta_{\text{new}}$.

Return $(u_1, \dots, u_N)$, the first N components of u. Note that the $(N + 1)$th component was merely the final Lagrange multiplier.

https://doi.org/10.1017/S1446181118000391 Published online by Cambridge University Press

TABLE 1. Values updated by $P_A = P_{\Omega_1 \times \cdots \times \Omega_6}$.

$P_{\Omega_1}(x_1)$	$P_{\Omega_2}(x_2)$	$P_{\Omega_3}(x_3)$	$P_{\Omega_4}(x_4)$	$P_{\Omega_5}(x_5)$	$P_{\Omega_6}(x_6)$
$\underline{x_{1_1}}$	$\underline{x_{2_1}}$	x_{3_1}	x_{4_1}	x_{5_1}	x_{6_1}
$\underline{x_{1_2}}$	$\underline{x_{2_2}}$	$\underline{x_{3_2}}$	x_{4_2}	x_{5_2}	x_{6_2}
x_{1_3}	$\underline{x_{2_3}}$	$\underline{x_{3_3}}$	$\underline{\underline{x_{4_3}}}$	x_{5_3}	x_{6_3}
x_{1_4}	x_{2_4}	$\underline{x_{3_4}}$	$\underline{\underline{x_{4_4}}}$	$\underline{\underline{x_{5_4}}}$	x_{6_4}
x_{1_5}	x_{2_5}	x_{3_5}	$\underline{\underline{x_{4_5}}}$	$\underline{\underline{x_{5_5}}}$	$\underline{\underline{x_{6_5}}}$
x_{1_6}	x_{2_6}	x_{3_6}	x_{4_6}	$\underline{\underline{x_{5_6}}}$	$\underline{\underline{x_{6_6}}}$

5.1. Alternative formulation We may also attempt to speed up convergence by considering two modified versions of the method. Consider the problem with a partition of seven segments, so $N = 6$. From the form of equation (2.3), for a single iteration, the values updated by an iteration $x \mapsto R_A(x)$ in the product space are underlined in Table 1. However, in the computation of the projection onto the agreement space (P_B) values are averaged across rows and so many unchanged values are included in the averaging step. More precise solutions require higher N and, for higher N, the ratio of unchanged values to changed values in the averaging step grows. This usually slows down computation and convergence. One possible solution is to reformulate the problem as a problem of computation with three sets, $\Omega_1 \cap \Omega_4, \Omega_2 \cap \Omega_5$ and $\Omega_3 \cap \Omega_6$, as detailed below. Here the updated values in each column which are underlined twice may be calculated separately from those underlined once and so this reformulation is no less amenable to parallelization. Notice that we can reformulate in this way for any $N > 3$ and that still only two unchanged values will remain at each step (one for the first partition point and one for the last). The memory necessary to store this product space vector x is smaller, although the number of projections computed remains the same, because the computation of $P_{\bigcap_k \Omega_{i+3k}}(x_i)$ requires the computation of $P_{\Omega_i}(x_i), P_{\Omega_{i+3}}(x_i), \ldots$ and so on.

Another approach is to simply change the map P_B so that only the changed row values are averaged in the agreement step. Then P_B is, in this reformulation, still a map into B. It is no longer the projection map, but we expect similar behaviour to that of the three-set reformulation, because the only difference is the inclusion or exclusion of two additional unchanged values for partition points 1 and N. Indeed, if we chose to include just two unchanged values—one for each of first and Nth partition points—then the formulations are equivalent. Thus, the altered P_B may be thought of as a map to *some near point* of the agreement set where the formulation in question is the three-set formulation. Because of this similarity, we do not consider these two approaches separately. For all of our examples we use the three-set reformulation which does not include unchanged values in the averaging step.

Simple code for computing the projections P_A and P_B may be seen in Algorithm 4. It uses the stored procedures from Algorithm 1. Note that the projection P_B is the

https://doi.org/10.1017/S1446181118000391 Published online by Cambridge University Press

TABLE 2. Reformulating as a three-set feasibility problem.

$P_{\Omega_1\cap\Omega_4}(x_1)$	$P_{\Omega_2\cap\Omega_5}(x_2)$	$P_{\Omega_3\cap\Omega_6}(x_3)$
$\underline{x_{1_1}}$	$\underline{x_{2_1}}$	x_{3_1}
$\underline{x_{1_2}}$	$\underline{x_{2_2}}$	$\underline{x_{3_2}}$
$\underline{\underline{x_{1_3}}}$	$\underline{x_{2_3}}$	$\underline{x_{3_3}}$
$\underline{\underline{x_{1_4}}}$	$\underline{x_{2_4}}$	$\underline{x_{3_4}}$
$\underline{\underline{x_{1_5}}}$	$\underline{\underline{x_{2_5}}}$	$\underline{\underline{x_{3_5}}}$
x_{1_6}	$\underline{\underline{x_{2_6}}}$	$\underline{\underline{x_{3_6}}}$

three-set reformulation described above, which does not include unchanged values in the averaging step.

Algorithm 4: Compute projection for $A = \Omega_1 \times \cdots \times \Omega_N$ and a near point in B.

procedure *Project onto A*
- **Data:** Receives as input a point $x = (x_1, \ldots, x_N) \in \mathbb{R}^{N\times N}$ ($x_k \in \mathbb{R}^N$ for all k)
- **Result:** Returns a point $u = (u_1, \ldots, u_N) \in \mathbb{R}^{N\times N}$ such that $u \in P_A(x)$.
- **for** $k \in \{1, \ldots, N\}$ **do**
  - set $u_k :=$ **Projection for** $\Omega_k(u_k)$;
- return u;

procedure *Concur in B*
- **Data:** Receives as input a point $x = (x_1, \ldots, x_N) \in \mathbb{R}^{N\times N}$ ($x_k \in \mathbb{R}^N$ for all k)
- **Result:** Returns a point $u = (\mu, \ldots, \mu) \in \mathbb{R}^{N\times N}$, where $\mu \in \mathbb{R}^N$ (clearly $u \in B$).
- set $\mu_1 := (x_{1,1} + x_{2,1})/2$;
- set $\mu_N := (x_{N-1,N} + x_{N,N})/2$;
- **for** $j \in \{2, \ldots, N-1\}$ **do**
  - set $\mu_j := (x_{j-1,j} + x_{j,j} + x_{j+1,j})/3$;
- **for** $k \in \{1, \ldots, N\}$ **do**
  - set $u_k := \mu$;
- return u;

6. Examples

For all of our examples, unless otherwise specified, we use as an initial point for the iterations $x_0 = (\omega, \ldots, \omega) \in R^{N\times N}$, where $\omega_i = \alpha + i(\beta - \alpha)/(N+1), i = 1, \ldots, N$, are the node values of the affine function satisfying the boundary values. We also use

https://doi.org/10.1017/S1446181118000391 Published online by Cambridge University Press

$N = 21$, unless otherwise specified. We compute the error in the natural way:

$$\epsilon = \frac{b-a}{N+1} \sum_{k=1}^{N} |\omega_k' - \omega_k|^2.$$

We will use the following terms when discussing the error.

(1) When ω_k' is the value of the true solution at $x_k = a + k(b-a)/(N+1)$, and ω_k represents the solution of the finite difference problem (2.3) at x_k calculated using Newton's method, ϵ measures the true error of the approximate solution from the true solution. We expect this error to decrease as N is increased. We show this error for each of our examples with both $N = 11$ and $N = 21$ in Table 5.
(2) When the ω_k are values obtained from DR or AP and the ω_k' are the values at x_k of the true solution, we call ϵ the *error from true solution*.
(3) When the ω_k are values obtained from DR or AP and the ω_k' are the values at x_k obtained by Newton's method (which is taken to be the numerical solution of the finite difference problem (2.3)), we call ϵ the *error from Newton solution*.
(4) For an iterate of AP, each iterate lies on the agreement set B and so we compute the error where the ω_k are the induced numerical solution. For each iterate of DR, we project back onto B to obtain a numerical solution. In either case, we take *relative error* to mean the change in numerical solution from one iterate to the next, the value of ϵ, when the w_k' and w_k values correspond to numerical solutions from the nth and $(n-1)$th steps of the method we are scrutinizing.

When we plot numerical solutions corresponding to various iterates of our methods (as at left in Figure 1), we report first the name of the method (DR or AP) followed by the number of the iterate for which we are plotting a numerical solution. We use the shorthand $N_E M := N \cdot 10^M$.

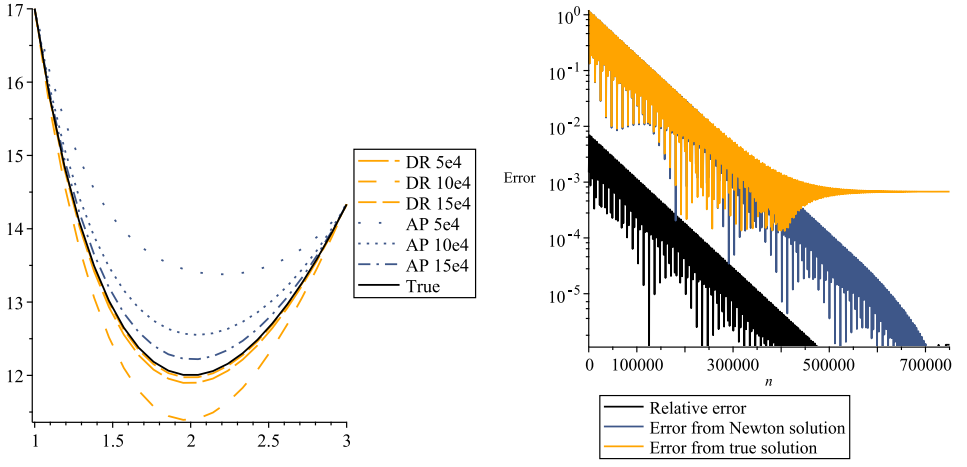
In cases where Newton's method converges, it generally achieves a difference between subsequent iterates of less than 10^{-12} within 10 steps. As will become apparent from the examples, this is so much faster than our methods as to render any comparison of speed useless. However, our methods sometimes work in cases where Newton's method struggles and they provide useful insights into the behaviour of such algorithms in the nonconvex setting more generally, complementing previous work in this area. The motivating and ideal conditions for implementation are further discussed in the conclusion (Section 7).

Example 6.1. We first tested the method on a simple problem from [12], namely, the differential equation $y'' = (32 + 2x^3 - yy')/8$ with boundary conditions $y(1) = 17$, $y(3) = 43/3$, which admits the smooth solution

$$y(x) = x^2 + 16/x.$$

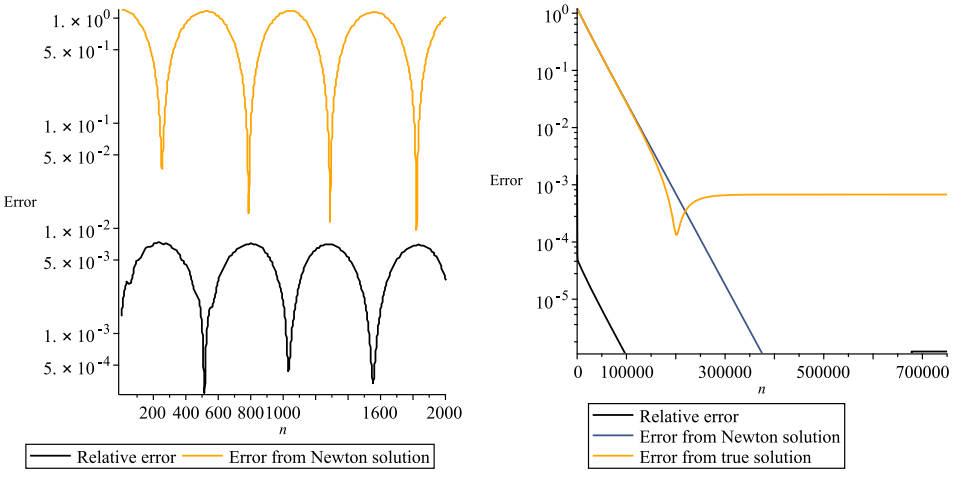
Here AP, DR and Newton's method all successfully solve the induced system of equations. Their behaviour is shown graphically in Figure 1, where $N = 21$.

https://doi.org/10.1017/S1446181118000391 Published online by Cambridge University Press



(a) True and numerical solutions (b) Error for DR iterates

FIGURE 1. Convergence behaviour for Example 6.1.



(a) DR (b) AP

FIGURE 2. Convergence behaviour for Example 6.1.

At around 400 000 iterates, the numerical solution from DR is close to the solution of the finite difference problem (2.3) and so the error from the true solution appears to stabilize, exposing the inherent error between the approximate solution (with 21 nodes) and the true solution.

Zooming in, the first 2000 iterates are shown at left in Figure 2; we see that the "solid" appearance in Figure 1 is created by shorter-scale oscillations in relative error.

https://doi.org/10.1017/S1446181118000391 Published online by Cambridge University Press

At right in Figure 2, we see the behaviour of AP, which converges more quickly and also without the drastic changes in relative error so typical of DR. This pattern of converging faster was observed often though not always, and the relative error plots for AP were similar in all our examples.

In the next two examples we consider the effect of partition size on the error from the true solution and on the rate of convergence.

Example 6.2. We consider the equation $y'' = -|y|$ with boundary conditions $y(-1) = 1, y(1) = -1$, which admits the smooth solution

$$y(x) = \begin{cases} c_1 \sin(x) + c_2 \cos(x), & x \le \frac{1}{2}\log\left(\frac{c_4}{e + c_4}\right), \\ c_3 \exp(x) + c_4 \exp(-x), & x > \frac{1}{2}\log\left(\frac{c_4}{e + c_4}\right), \end{cases}$$

$$c_1 = \frac{c_2 \cos(1) - 1}{\sin(1)},$$

$$c_2 = \frac{-\{\tan(1) + \tan(\frac{1}{2}\log(c_4/(\mathrm{e} + c_4)))\}}{\tan(1)\tan(\frac{1}{2}\log(c_4/(\mathrm{e} + c_4)))\sin(1) - \cos(1)\tan(1) - \cos(1)\tan(\frac{1}{2}\log(c_4/(\mathrm{e} + c_4))) - \sin(1)},$$

$$c_3 = -\frac{c_4 \mathrm{e}^{-1} + 1}{\mathrm{e}},$$

$$c_4 \approx 0.6453\,425\,944.$$

We found convergence for each of our methods. The true solution and the effect of partition fineness (N) on the error between various approximations and the true solution are shown at left in Figure 3.

Example 6.3. We examine the differential equation

$$y'' = \begin{cases} 0, & x < 0, \\ y, & x \ge 0 \end{cases}$$

with boundary conditions $y(-1) = -1$ and $y(1) = 1$, which admits the smooth solution

$$y(x) = \begin{cases} \left(\frac{\mathrm{e}^{-1} + 2}{2\mathrm{e}} + \frac{1}{2}\right)x + \left(\frac{\mathrm{e}^{-1} + 2}{2\mathrm{e}} - \frac{1}{2}\right), & x < 0, \\ \frac{\mathrm{e}^{-1} + 2}{2\mathrm{e}}\mathrm{e}^{x} - \frac{1}{2}\mathrm{e}^{-x}, & x \ge 0. \end{cases}$$

The true solution and the effect of N on the error between true and approximate solutions is shown at right in Figure 3. A convergence plot for DR is given in Figure 4, where $N = 11$ is shown at left and $N = 21$ is shown at right.

Noting the different horizontal axis scales, it may be seen that, as one would expect, convergence is much more rapid for smaller N, a phenomenon which held both consistently and dramatically across all our examples.

https://doi.org/10.1017/S1446181118000391 Published online by Cambridge University Press

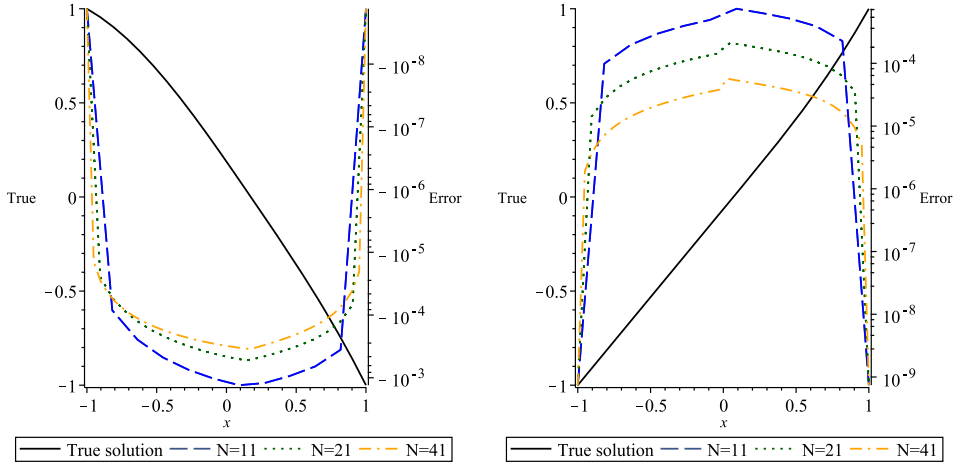


FIGURE 3. True solutions (left axis scale) and effect of partition size on error between true solution and estimate by Newton (right axis scale) for Examples 6.2 (left) and 6.3 (right).

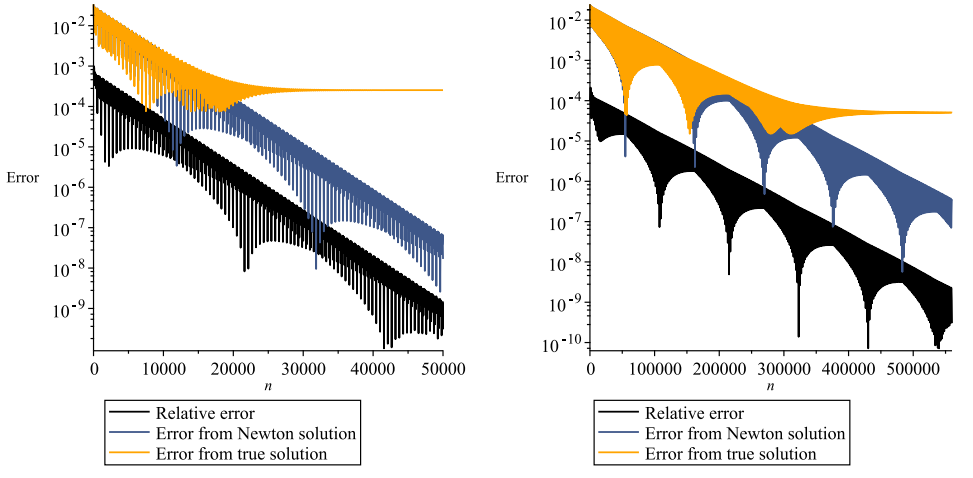


FIGURE 4. Effect of N on DR convergence for Example 6.3.

The "aqueducts"—which might seem to suggest long-scale oscillations in the change from iterate to iterate—appear to be an artefact of the sample of iterates we used to prepare the plot. For $N = 21$ our plot is made from sampling at every 400th iterate. Shorter-scale oscillations of the kind visible in Figure 2 appeared for all of our error plots and, by sampling infrequently, we tend to sample near the tops and sides of the humps while missing the valleys. This phenomenon combined with the regularity of the shorter-scale oscillations creates the illusion of aqueducts.

https://doi.org/10.1017/S1446181118000391 Published online by Cambridge University Press

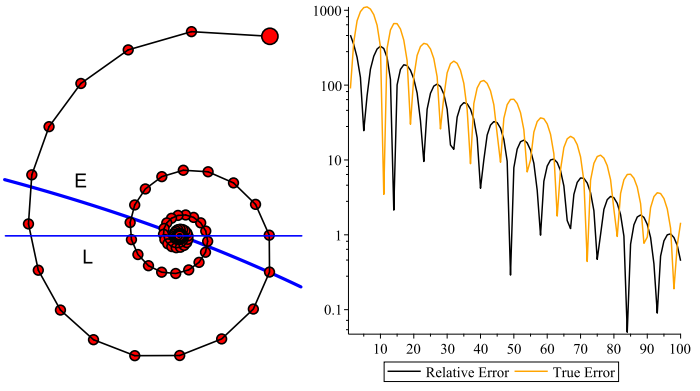


FIGURE 5. Relative error and error from true solution for converging DR iterates for an ellipse E and a line L.

The relative error plots do, however, reveal an important characteristic of the behaviour. The change in error from the true solution does *not* track the relative error between iterates, but *instead* roughly tracks the change in relative error at the tops of the humps in Figure 2. Once sufficiently close to the solution, these oscillations become regular and so convergence can be estimated by tracking only the iterates where relative error peaks.

This behaviour is consistent with the behaviour of DR in other contexts. At left in Figure 5, we see DR iterates for an ellipse and a line. The line is the analogue of our diagonal set B (3.1) and so at right we report $\|P_L x_{n+1} - P_L x_n\|_2$. The similarities to Figure 2 are unmistakable.

In each of the next three examples we consider the sensitivity of the methods to the starting point. For the first two examples we have multiple potential solutions and for the final example Newton's method may cycle rather than finding a solution.

EXAMPLE 6.4. The differential equation $y'' = -|y|$ with boundary conditions $y(0) = 0, y(4) = -2$ admits two possible smooth solutions:

$$y_1(x) = -\frac{2\sinh(x)}{\sinh(4)}, \tag{6.1}$$

$$y_2(x) = \begin{cases} \dfrac{2\sin(x)}{\sinh(4-\pi)}, & x \le \pi, \\ -\dfrac{2\sinh(\pi - x)}{\sinh(\pi - 4)}, & x > \pi. \end{cases} \tag{6.2}$$

Here we initially found convergence for small N, but our scripts stopped working for larger N. Investigating, we found that MAPLE's FSOLVE was unable to compute the solution to the Lagrangian system for the P_{Ω_i}. Replacing it with our own numerical solver, we recovered convergence. With the starting values corresponding to the affine function matching the boundary conditions, all methods converged to the solution y_1

https://doi.org/10.1017/S1446181118000391 Published online by Cambridge University Press

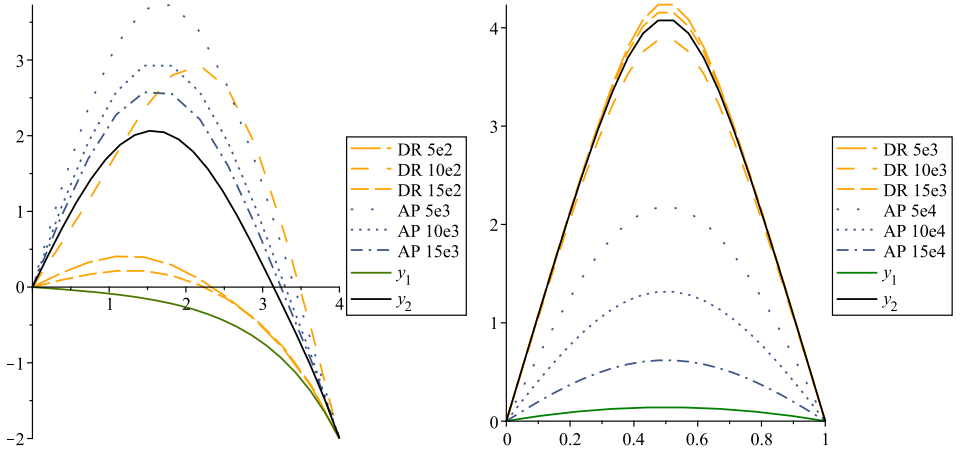


FIGURE 6. DR and AP may converge to two different solutions from the same starting point: at left, Example 6.4; at right, Example 6.5.

TABLE 3. Sensitivity to starting point for Example 6.4: 1 or 2 indicate that the method converged to y_1 or y_2, respectively, while S indicates that the method appeared stuck after 5E5 iterates.

Method/Start λ	0.01	0.1	0.5	1	2	3	4	5	6	7	8	9
Newton $N = 11$	2	2	2	2	2	2	2	2	2	2	2	2
DR $N = 11$	1	1	2	2	2	2	1	1	1	1	1	1
AP $N = 11$	1	1	2	2	2	2	2	2	2	2	2	2
Newton $N = 21$	2	2	2	2	2	2	2	2	2	2	2	2
DR $N = 21$	1	1	2	2	2	S	S	S	S	2	2	2
AP $N = 21$	1	1	2	2	2	2	2	2	2	2	2	2
Method/Start λ	−0.01	−0.1	−0.5	−1	−2	−3	−4	−5	−6	−7	−8	−9
Newton $N = 11$	1	1	1	1	1	1	1	1	1	1	1	1
DR $N = 11$	1	1	1	1	1	1	1	1	1	1	1	1
AP $N = 11$	1	1	1	1	1	1	1	1	1	1	1	1
Newton $N = 21$	1	1	1	1	1	1	1	1	1	1	1	1
DR $N = 21$	1	1	1	1	1	S	S	S	S	2	2	2
AP $N = 21$	1	1	1	1	1	1	1	1	1	1	1	1

from (6.1). However, with the starting values matching the boundary conditions and $4\chi_{(0,1)}$ everywhere else, AP goes to the "nearer" solution of y_2 (6.2), while DR finds its way down to y_1. This may be seen in Figure 6.

We repeated the experiment for a variety of starting points corresponding to functions which matched the boundary values and were $\lambda\chi_{(0,4)}$ everywhere else for various λ. The results are tabulated in Table 3.

https://doi.org/10.1017/S1446181118000391 Published online by Cambridge University Press

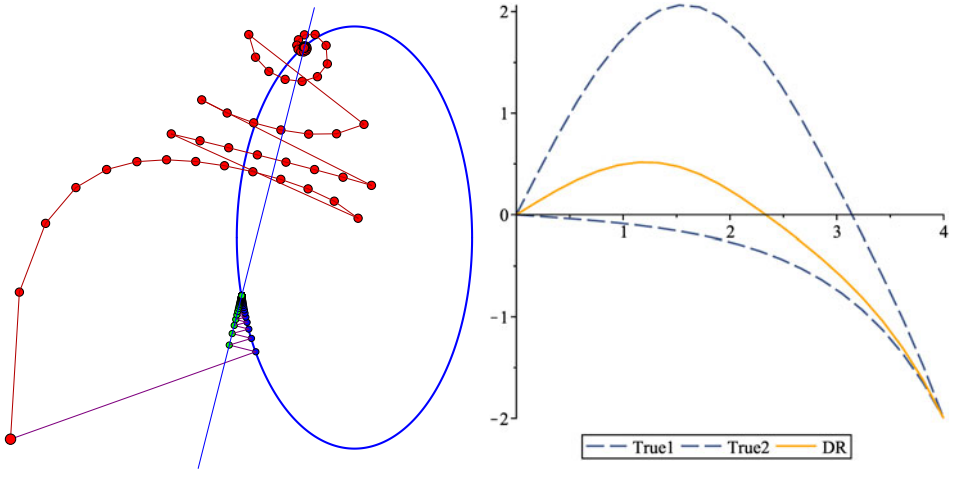


FIGURE 7. DR started sufficiently far from two feasible points may converge to the farther of the two while AP converges to the nearer (left). For Example 6.4, after 5E5 iterates, DR appears stuck for some starting points (right).

Newton's method behaved very predictably, always converging to y_1 for $\lambda < 0$ and y_2 for $\lambda > 0$ regardless of partition size. AP was slightly less predictable, converging to y_1 for $\lambda = 0.01$. For $\lambda = 0.1$, it appeared stuck between y_1 and y_2 even after 15E4 iterates regardless of partition size; eventually it converged to y_1.

The behaviour of DR, by contrast, was highly unpredictable, changed drastically with partition size and frequently converged to the "farther" away of the two solutions, when started some distance from both. This is consistent with the known behaviour of DR illustrated in Figure 7 (see, for example, the article by Borwein et al. [8]).

We observed another trend as well. For $\lambda = 4$ and $N = 11$, DR converged to y_1 while for $\lambda = 2$ it converged to y_2; for $\lambda = 3$, convergence was extremely delayed. For most values, we were able to ascertain the eventual solution within 15E4 iterates. For some λ values we were unable to tell even after 5E5 iterates. This pattern of "crossroad" points taking longer to close on a destination held consistently. One example is shown at right in Figure 7.

EXAMPLE 6.5. The differential equation $y'' = -\exp(y)$ with boundary conditions $y(0) = y(1) = 0$ admits two smooth solutions:

$$y(x) = \log\left(c - c\tanh^2\left(\sqrt{\frac{c}{2}}(1/2 - x)\right)\right),$$

$$\text{where } c \approx 1.1508 \tag{6.3}$$

$$\text{or } c \approx 59.827. \tag{6.4}$$

When the starting values match the unique affine function corresponding to the boundary conditions, all of the numerical methods converge to the particular solution

https://doi.org/10.1017/S1446181118000391 Published online by Cambridge University Press

TABLE 4. Sensitivity to starting point for Example 6.5: 1, 2 indicate that the method converged to y_1, y_2, while "D" and "S", respectively, indicate that the method diverged or appeared to hover.

Method/Start λ	-1	0	1	2	3	4	5	6	7
Newton $N = 11$	1	1	1	1	2	D	D	D	D
DR $N = 11$	1	1	1	2	2	2	2	2	2
AP $N = 11$	1	1	1	1	2	2	2	S	S
Newton $N = 21$	1	1	1	1	2	D	D	D	D
DR $N = 21$	1	1	1	2	2	2	2	2	2
AP $N = 21$	1	1	1	1	2	2	2	2	S

given by (6.3), which we call y_1. If we start instead from a function matching the boundary conditions and $2\chi_{(0,1)}$ everywhere else, for $N = 21$ AP still goes to y_1 while DR converges to the other solution y_2 given by (6.4). This can be seen in Figure 6.

We again repeated the experiment for a variety of starting points corresponding to functions which matched the boundary values and were $\lambda\chi_{(0,1)}$ elsewhere for various λ. The results are tabulated in Table 4, where it may be seen that for certain starting values Newton's method diverged or AP appeared stuck after 15E4 iterates.

EXAMPLE 6.6. We consider the second order differential equation

$$y''(x) = \begin{cases} -1, & y(x) < 0, \\ 1, & y(x) \geq 0 \end{cases}$$

together with the boundary conditions $y(-1) = -1$ and $y(1) = 1$.

Here the right-hand side, being a Heaviside function [27], fails to satisfy the standard conditions for existence and uniqueness. Nonetheless, it is readily seen to admit a unique continuous solution on the interval $[-1, 1]$, namely the odd function

$$y(x) = \begin{cases} -\frac{1}{2}x^2 + \frac{1}{2}x, & x < 0, \\ \frac{1}{2}x^2 + \frac{1}{2}x, & x \geq 0. \end{cases}$$

This example is especially interesting, because while Newton's method finds the solution when starting from the affine function satisfying the boundary criteria, it fails to converge to the solution when started at $1\chi_{(-1,1)}, \ldots, 7\chi_{(-1,1)}$. Instead, it cycles between the two nonsolutions shown at left in Figure 8.

Within six iterates of Newton's method, the norm of the difference between subsequent even iterates or subsequent odd iterates is less than 1E−19. By way of contrast, DR and AP appear to work well from all of these starting points. At right we show a plot of relative error for DR with 21 iterates starting from the affine function values.

We provide an overview summary of our experimental results in Table 5. In the first column we report how many iterates it took for $\log_{10}$ of the "peak" relative error

https://doi.org/10.1017/S1446181118000391 Published online by Cambridge University Press

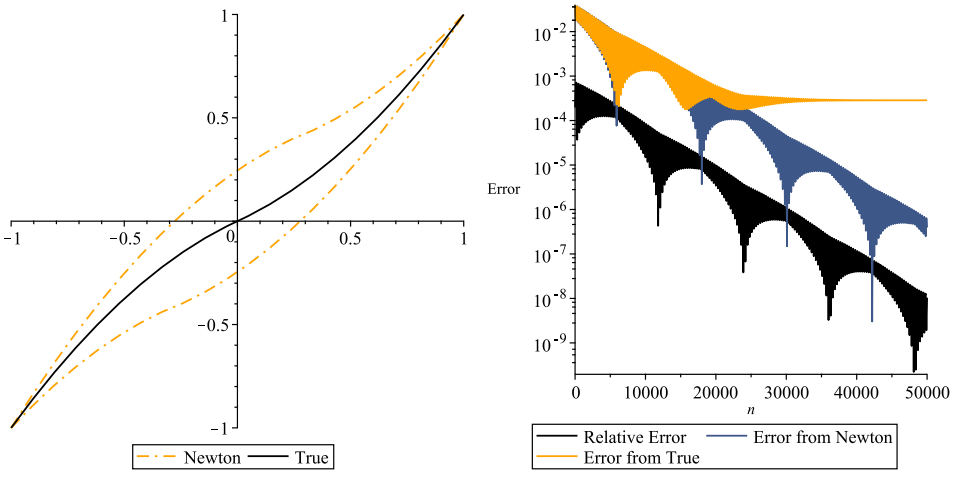


FIGURE 8. Newton's method may cycle for certain starting points in Example 6.6 (left), while DR converges (right).

TABLE 5. A summary of experimental results from all examples.

	DR 1E–1	AP 1E–1	DR wave	DR error/relative	AP error/relative	True error
Example 6.1 $N = 11$	9E3	4E3	142	44	2E3	3.4E–3
$N = 21$	129E3	60E3	516	155	26E3	6.7E–4
Example 6.2 $N = 11$	18E3	9E3	198	63	4E3	4.7E–4
$N = 21$	247E3	102E3	715	227	53E3	1.3E–4
Example 6.3 $N = 11$	9E3	4E3	138	43	2E3	2.5E–4
$N = 21$	117E3	58E3	500	155	25E3	5.1E–5
Example 6.4 $N = 11$	2E3	1E3	65	19	4E2	3.1E–3
$N = 21$	25E3	12E3	230	67	5E3	6.2E–4
Example 6.5 $N = 11$	16E3	8E3	184	57	34E2	2.6E–5
$N = 21$	208E3	104E3	670	211	46E3	5.1E–6
Example 6.6 $N = 11$	1E3	4E2	41	12	1E2	1.4E–3
$N = 21$	11E3	5E3	149	46	2E3	2.9E–4

for DR to go down by 1. In the second column we report this for AP where peaks need no longer be considered. In the third column we give the average number of iterates which compose the individual oscillations in the relative error of DR (as in Figure 2). In the fourth column we report for DR the ratio of peak *error from the approximate solution* to peak *relative error*. Because the two different peaks do not

https://doi.org/10.1017/S1446181118000391 Published online by Cambridge University Press

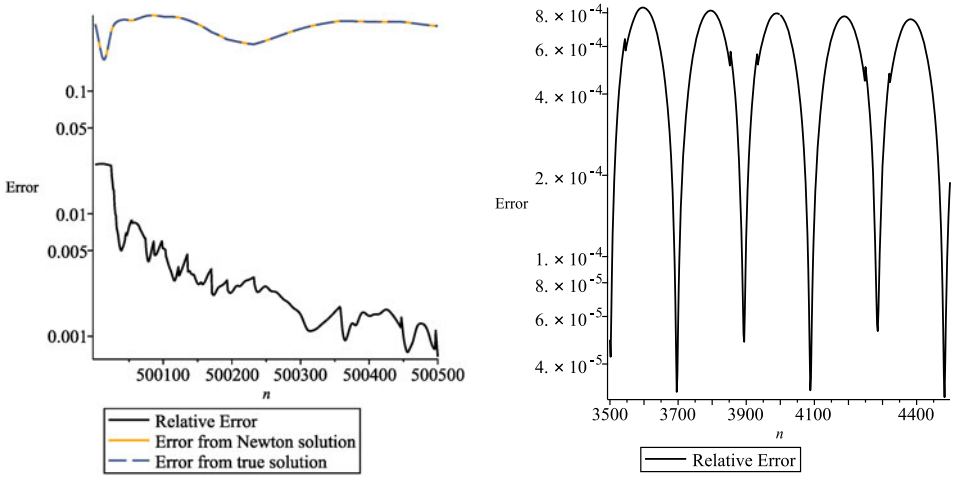


FIGURE 9. Stuck DR (left); relative error tends toward a pattern other than smooth oscillation (right).

coincide, we take each peak in the error from the approximate solution and compare it to the *previous* peak in the relative error. In the fifth column we report for AP the ratio of error from the approximate solution to the relative error; in this case peaks no longer need be considered. In the final column we show the error of the approximate solution (2.3) (obtained by Newton's method) from the true solution.

Analysis of a stuck problem revealed that regular oscillations in relative error were conspicuously absent. This is shown at left in Figure 9, where for Example 6.4 with $N = 21$ and starting with $\lambda = 6$ DR appears stuck after 5E5 iterates. Original attempts to catalogue average oscillation length for relative error resulted in data which appeared at times periodic. This led to the discovery that the pattern in relative error may tend toward a predictable pattern other than smooth oscillation. This is shown for Example 6.2 with $N = 11$ at right in Figure 9.

7. Conclusion

The poor trade-off in convergence rate for finer partitions suggests some modifications to the method for solving real-world problems. One such modification is to begin with a coarse partition and increase the fineness over time. Another is to simply switch to a more traditional solver once sufficient proximity to the true solution is suspected from analysis of the relative error.

The impressive stability of DR relative to more traditional methods is consistent with previous findings in the application of these methods to finding the intersections of analytic curves [21]. This property and its unique suitability for parallelization make it an ideal candidate for employment in settings where traditional solvers fail or for getting close enough to a solution that traditional solvers may be applied.

https://doi.org/10.1017/S1446181118000391 Published online by Cambridge University Press

Acknowledgements

The authors wish to thank the anonymous referees for their careful reading and detailed, helpful feedback.

References

[1] F. J. Aragón Artacho and J. M. Borwein, "Global convergence of a non-convex Douglas–Rachford iteration", *J. Global Optim.* **57** (2013) 753–769; doi:10.1007/s10898-012-9958-4.

[2] F. J. Aragón Artacho, J. M. Borwein and M. K. Tam, "Recent results on Douglas–Rachford methods for combinatorial optimization problems", *J. Optim. Theory Appl.* **163** (2014) 1–30; doi:10.1007/s10957-013-0488-0.

[3] F. J. Aragón Artacho, J. M. Borwein and M. K. Tam, "Douglas–Rachford feasibility methods for matrix completion problems", *ANZIAM J.* **55** (2014) 299–326; doi:10.1017/S1446181114000145.

[4] F. J. Aragón Artacho and R. Campoy, "Solving graph coloring problems with the Douglas–Rachford algorithm", *Set-Valued Var. Anal.* **26** (2018) 277–304; doi:10.1007/s11228-017-0461-4.

[5] H. H. Bauschke and P. L. Combettes, *Convex analysis and monotone operator theory in Hilbert spaces*, 2nd edn (Springer, Cham, 2017); doi:10.1007/978-3-319-48311-5.

[6] H. H. Bauschke, P. L. Combettes and D. R. Luke, "Phase retrieval, error reduction algorithm, and Fienup variants: a view from convex optimization", *J. Opt. Soc. Amer.* A **19** (2002) 1334–1345; doi:10.1364/JOSAA.19.001334.

[7] J. Benoist, "The Douglas–Rachford algorithm for the case of the sphere and the line", *J. Global Optim.* **63** (2015) 363–380; doi:10.1007/s10898-015-0296-1.

[8] J. M. Borwein, S. B. Lindstrom, B. Sims, A. Schneider and M. P. Skerritt, "Dynamics of the Douglas–Rachford method for ellipses and p-spheres", *Set-Valued Var. Anal.* **26** (2018) 385–403; doi:10.1007/s11228-017-0457-0.

[9] J. M. Borwein and B. Sims, "The Douglas–Rachford algorithm in the absence of convexity", in: *Fixed-point algorithms for inverse problems in science and engineering*, Volume 49 of *Springer Optim. Appl.* (Springer, New York, 2011) 93–109; doi:10.1007/978-1-4419-9569-8_6.

[10] J. M. Borwein and M. K. Tam, "Reflection methods for inverse problems with applications to protein conformation determination", Springer volume on the CIMPA school *Generalized Nash equilibrium problems, bilevel programming and MPEC*, New Delhi, India, December 2012; doi:10.1007/978-981-10-4774-9_5.

[11] J. M. Borwein and M. K. Tam, "A cyclic Douglas–Rachford iteration scheme", *J. Optim. Theory Appl.* **160** (2014) 1–29; doi:10.1007/s10957-013-0381-x.

[12] R. L. Burden, D. J. Faires and A. M. Burden, *Numerical analysis* (Cengage Learning, Boston, MA, 2016).

[13] M. N. Dao and H. M. Phan, "Linear convergence of projection algorithms", Preprint, 2016, arXiv:1609.00341.

[14] M. N. Dao and M. K. Tam, "A Lyapunov-type approach to convergence of the Douglas–Rachford algorithm", *J. Global Optim.* **73** (2019) 83–112; doi:10.1007/s10898-018-0677-3.

[15] J. Douglas and H. H. Rachford, "On the numerical solution of the heat conduction problem in 2 and 3 space variables", *Trans. Amer. Math. Soc.* **82** (1956) 421–439; doi:10.2307/1993056.

[16] V. Elser, I. Rankenburg and P. Thibault, "Searching with iterated maps", *Proc. Natl. Acad. Sci. USA* **104** (2007) 418–423; doi:10.1073/pnas.0606359104.

[17] S. Gravel and V. Elser, "Divide and concur: a general approach to constraint satisfaction", *Phys. Rev.* E **78** (2008) 036706; doi:10.1103/PhysRevE.78.036706.

[18] H. B. Keller, *Numerical methods for two-point boundary value problems* (Blaisdell, Waltham, MA, 1968).

[19] A. S. Lewis, D. R. Luke and J. Malick, "Local linear convergence for alternating and averaged nonconvex projections", *Found. Comput. Math.* **9** (2009) 485–513; doi:10.1007/s10208-008-9036-y.

[20] S. B. Lindstrom and B. Sims, "Survey: sixty years of Douglas–Rachford", Preprint, 2018, arXiv:1809.07181.
[21] S. B. Lindstrom, B. Sims and M. P. Skerritt, "Computing intersections of implicitly specified plane curves", *J. Nonlinear Convex Anal.* **18** (2017) 347–359; http://www.ybook.co.jp/online2/opjnca/vol18/p347.html.
[22] P. L. Lions and B. Mercier, "Splitting algorithms for the sum of two nonlinear operators", *SIAM J. Numer. Anal.* **16** (1979) 964–979; doi:10.1137/0716071.
[23] H. M. Phan, "Linear convergence of the Douglas–Rachford method for two closed sets", *Optimization* **65** (2016) 369–385; doi:10.1080/02331934.2015.1051532.
[24] G. Pierra, "Decomposition through formalization in a product space", *Math. Program.* **28** (1984) 96–115; https://link.springer.com/article/10.1007/BF02612715.
[25] A. W. Roberts and D. E. Varberg, *Convex functions* (Academic Press, New York, 1973).
[26] B. F. Svaiter, "On weak convergence of the Douglas–Rachford method", *SIAM J. Control Optim.* **49** (2011) 280–287; doi:10.1137/100788100.
[27] E. W. Weisstein, "Heaviside step function", *Wolfram MathWorld*, Wolfram Web Resources; http://mathworld.wolfram.com/HeavisideStepFunction.html.