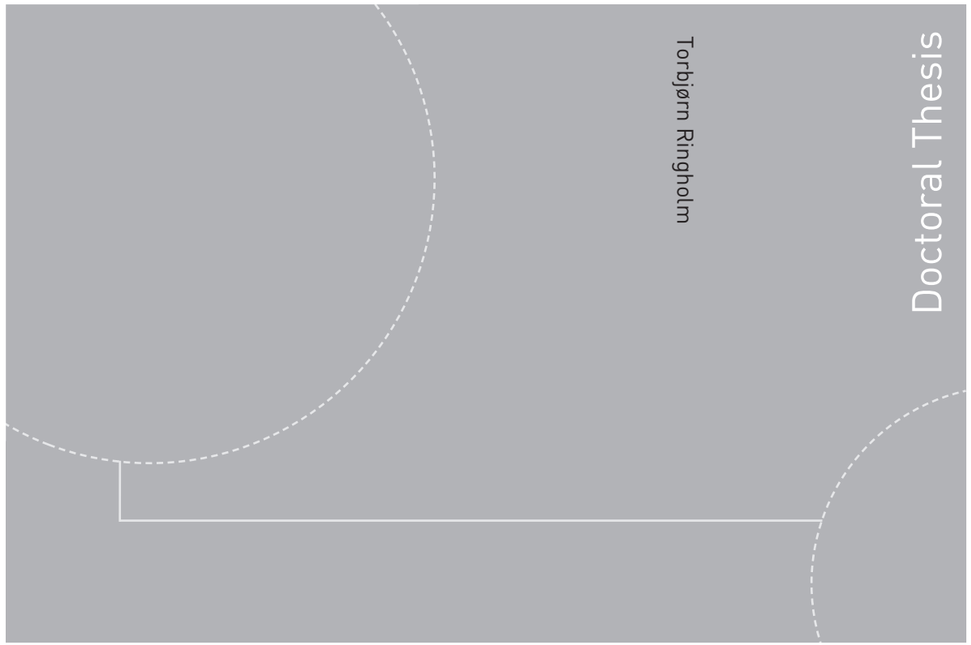
ISBN 978-82-326-3260-2 (printed version)
ISBN 978-82-326-3261-9 (electronic version)
ISSN 1503-8181



Doctoral theses at NTNU, 2018:235

Torbjørn Ringholm

Discrete gradient methods in image processing and partial differential equations on moving meshes

NTNU
Norwegian University of
Science and Technology
Faculty of Information Technology
and Electrical Engineering
Department of Mathematical Sciences

NTNU
Norwegian University of
Science and Technology

Doctoral theses at NTNU, 2018:235

NTNU

NTNU
Norwegian University of
Science and Technology

Torbjørn Ringholm

Discrete gradient methods in image processing and partial differential equations on moving meshes

Thesis for the degree of Philosophiae Doctor

Trondheim, August 2018

Norwegian University of Science and Technology
Faculty of Information Technology
and Electrical Engineering
Department of Mathematical Sciences

NTNU
Norwegian University of
Science and Technology

NTNU
Norwegian University of Science and Technology

Thesis for the degree of Philosophiae Doctor

Faculty of Information Technology
and Electrical Engineering
Department of Mathematical Sciences

© Torbjørn Ringholm

ISBN 978-82-326-3260-2 (printed version)
ISBN 978-82-326-3261-9 (electronic version)
ISSN 1503-8181

Doctoral theses at NTNU, 2018:235

Printed by Skipnes Kommunikasjon as

Preface

This thesis is submitted in partial fulfilment of the requirements for the degree of philosophiae doctor (PhD) at the Norwegian University of Science and Technology (NTNU). The research was funded by the Department of Mathematical Sciences at NTNU. The work was carried out at NTNU, with the exception of two research stays at the Department of Applied Mathematics and Theoretical Physics at the University of Cambridge. Additional funding has been provided by the EU Horizon 2020 project "Challenges in preserving structures" and the RCN research project "Structure preserving integrators, discrete integrable systems and algebraic combinatorics".

The four years of work represented by this thesis has been the most interesting and rewarding, yet also humbling and taxing, experience of my life so far and I am not certain that I would have finished had it not been for the help and support of those around me. First of all I would like to direct my thanks to my supervisor, Brynjulf Owren, who with his sensible advice and infectious enthusiasm has kept me on course and motivated throughout this period. I would also like to thank my co-supervisor Elena Celledoni for her insightful comments, and for giving me the opportunity to visit Cambridge. Moreover, I would like to thank my co-authors Matthias Ehrhardt, Sølve Eidnes, Jasmina Lazić, Erlend Riis and Carola-Bibiane Schönlieb for their hard work and patience. Furthermore, I would like to extend my gratitude to all my coworkers and friends at NTNU, who have made this journey all the more pleasant and noteworthy.

On a personal level I would like to thank Vanje Rebni Kjer for her love and support and patience, my parents Toril Ringholm and Øyvind Sundheim for their encouragement, and my siblings Magnus Ringholm and Rebekka Ringholm for their valuable advice and optimism.

Torbjørn Ringholm
Trondheim, July 27, 2018

Contents

Introduction

Occurring in all branches of natural science, the study of differential equations is one of the main pursuits of mathematics and has been so since their independent inventions by Newton and Leibniz. Differential equations describe the dynamics of many systems, and in many cases the systems are too complicated to describe in a closed-form solution. Therefore, approximative methods for solving differential equations have an almost equally long history [13]. Spurred by the rise of electronic computing in the 20th century, the study of numerical solutions to differential equations has produced indispensable tools for society in the form of algorithms which we will term *numerical integrators*, not to be confused with methods for evaluating integrals. Numerical integrators are used for the simulation of systems occurring in many fields such as physics, chemistry, finance and biology.

Until the 1980s, research in numerical integration was geared toward producing accurate general-purpose solvers that could be applied to a wide range of problems with little user effort; the successes of this endeavour include Runge-Kutta methods, multistep methods, finite difference methods and finite element methods [44]. Over the past three decades, there has been great interest in developing more specialized integrators tailored to smaller classes of numerical integration problems that have geometric invariants as described by differential geometry. Examples of such invariants include energy preservation [32], volume preservation [48], symplecticity [27], isospectrality [2], and Lie group structure [21]. Since these properties manifest themselves in different ways, the numerical integration schemes designed to exploit them are of an equally diverse nature, with a specific scheme for each type of structure. The study and use of the subset of numerical integration methods designed to preserve geometric structure is termed *geometric numerical integration* [17].

The recurring theme of this thesis is the geometric numerical integration of ordinary differential equations (ODEs) and partial differential equations (PDEs) that exhibit conservation or dissipation of an energy, and the main tools of geometric numerical integration we shall employ are the soon-to-be-introduced discrete gradient methods. The discrete gradient methods are applied in various settings, and so the thesis can be roughly considered as consisting of three

topics, the last two of which overlap to a certain degree.

1. Energy preserving time stepping for spatially adaptive discretizations of PDEs.

2. The use of discrete gradient schemes in optimization methods, with applications in image analysis.

3. Generalizations of discrete gradients to dissipative and conservative problems on Riemannian manifolds, with applications in numerical optimization.

In the following we will briefly review the concepts used in the papers that constitute the main scientific contribution of this thesis.

1.1 Discrete gradient schemes for ODEs

An initial value problem in $\mathbb{R}^n$ consists of finding a curve $u : \mathbb{R} \to \mathbb{R}^n$ such that

$$\dot{u} = f(u), \quad u(0) = u_0 \tag{1.1.1}$$

where $\dot{u}$ denotes the derivative of $u(t)$ and $f : \mathbb{R}^n \to \mathbb{R}^n$ is typically assumed to be Lipschitz continuous. A differentiable function $V : \mathbb{R}^n \to \mathbb{R}$ is said to be a *first integral* of (1.1.1) if for any solution $u(t)$ of (1.1.1),

$$\frac{\mathrm{d}}{\mathrm{d}t} V(u) = \nabla V(u)^T \dot{u} = \nabla V(u)^T f(u) = 0.$$

From the above, it is clear that if u solves (1.1.1), then $V(u(t)) = V(u(0))$ for all t, and so a first integral is a conserved quantity under the flow of (1.1.1). As an example of first integral preserving systems, many ODEs arising from physics conserve at least one form of energy, such as the Hamiltonian in Hamiltonian systems [28]. This kind of first integrals is so ubiquitous in the literature that the terms *first integral* and *energy* are used interchangeably, a convention that will be used in the following. Similar to the conservative ODEs, there exist ODEs that dissipate an energy, i.e. where $V(t)$ is decreasing:

$$\frac{\mathrm{d}}{\mathrm{d}t} V(u) = \nabla V(u)^T f(u) \leq 0.$$

As shown in [32], an ODE that conserves an energy V can be written on the form

$$\dot{u} = S(u) \nabla V(u), \tag{1.1.2}$$

where $S(u)$ is a skew-symmetric matrix, under mild assumptions. The corresponding form for dissipative systems is

$$\dot{u} = A(u)\nabla V(u), \tag{1.1.3}$$

where $A(u)$ is a symmetric negative definite matrix. Systems of type (1.1.2) or (1.1.3) can be solved numerically by the use of *discrete gradient* schemes that retain the conservation or dissipation properties of the original ODE. A discrete gradient of an energy V is a continuous function $\overline{\nabla} V : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$ satisfying the two conditions

$$\left\langle \overline{\nabla} V(u, v), u - v \right\rangle = V(u) - V(v)$$
$$\overline{\nabla} V(u, u) = \nabla V(u),$$

where $\langle \cdot, \cdot \rangle$ denotes the Euclidian inner product. By means of a discrete gradient, we can construct a scheme for solving (1.1.2):

$$u^{k+1} - u^k = \tau \overline{S}(u^{k+1}, u^k)\overline{\nabla} V(u^{k+1}, u^k), \tag{1.1.4}$$

where $k \in \mathbb{N}$ is the iteration number, τ is the step size, u^k denotes the numerical approximation of $u(k\tau)$ and $\overline{S}(u^{k+1}, u^k)$ is a skew-symmetric, consistent approximation to S, meaning $\overline{S}(u, u) = S(u)$. The above scheme preserves V since

$$\begin{aligned} V(u^{k+1}) - V(u^k) &= \left\langle \overline{\nabla} V(u^{k+1}, u^k), u^{k+1} - u^k \right\rangle \\ &= \tau \left\langle \overline{\nabla} V(u^{k+1}, u^k), \overline{S}(u^{k+1}, u^k)\overline{\nabla} V(u^{k+1}, u^k) \right\rangle \\ &= 0. \end{aligned}$$

Note that the same result holds if τ is allowed to vary from iteration to iteration, which allows for the use of adaptive time steps. Similarly, a discrete gradient scheme for (1.1.3) of the form

$$u^{k+1} - u^k = \tau \overline{A}(u^{k+1}, u^k)\overline{\nabla} V(u^{k+1}, u^k), \tag{1.1.5}$$

where $\overline{A}$ is a symmetric negative definite matrix approximating A, preserves dissipativity in the sense that

$$V(u^{k+1}) \leq V(u^k).$$

There exist a number of discrete gradient functions, the three most popular of which are the mean value, or Average Vector Field (AVF) discrete gradient [18]

$$\overline{\nabla} V_{\text{AVF}}(u, v) = \int_0^1 \nabla V(\xi u + (1 - \xi) v) \mathrm{d}\xi,$$

the midpoint (MP), or Gonzalez, discrete gradient [14]

$$\overline{\nabla} V_{\mathrm{MP}}(u,v) = \nabla V\left(\frac{u+v}{2}\right) + \frac{V(v) - V(u) - \nabla V((v+u)/2)^T (v-u)}{\|v-u\|^2}(v-u)$$

and the Itoh-Abe discrete gradient [22] which can be expressed coordinate-wise as

$$\overline{\nabla} V_{\mathrm{IA}}(u,v)_l = \frac{V(\sum_{i=1}^{l} v_i e_i + \sum_{i=l+1}^{n} u_i e_i) - V(\sum_{i=1}^{l-1} v_i e_i + \sum_{i=l}^{n} u_i e_i)}{v_l - u_l} e_l$$

where e_l denotes Euclidean unit vector number l. Discrete gradient schemes are one-step integrators, meaning that they require knowledge of u^k only to compute the next step u^{k+1}, as opposed to multistep schemes that require knowledge of u^{k-r}, $r = 0, 1, .., s$, for an s-step scheme.

The AVF and MP discrete gradients give rise to symmetric schemes via (1.1.4) and (1.1.5) provided $\overline{S}(u,v) = \overline{S}(v,u)$ and $\overline{A}(u,v) = \overline{A}(v,u)$, respectively. These discrete gradients therefore yield second-order schemes, meaning that the u^k produced by (1.1.4) and (1.1.5) are $\mathcal{O}(\tau^2)$ approximations to $u(k\tau)$. The Itoh-Abe discrete gradient is first-order accurate, providing $\mathcal{O}(\tau)$ approximations; it can be symmetrized by composition with its adjoint, in which case one obtains a second-order scheme. It is worth noting that discrete gradient methods can also be applied to systems with multiple preserved energies. Strategies for doing so are detailed in [32] and [8].

Common to all the above discrete gradients is that they are implicit in the sense that they give rise to numerical schemes of the form

$$u^{k+1} = F(u^k, u^{k+1}), \tag{1.1.6}$$

necessitating the solution of a system of, generally, non-linear equations to obtain the iteration u^{k+1}. This is typically not a desirable trait as it slows down the computational speed of the scheme considerably and so it is of interest to reduce the cost of solving this system. If F is linear, one only needs to solve a linear system at each iteration, reducing the cost considerably from the non-linear case; also, if the problem of solving (1.1.6) can be decomposed into several smaller problems, one can distribute the workload over several computers working in parallel for additional speedup.

1.2 Conservative methods for PDEs and adaptivity

A first-order-in-time partial differential equation can, in general, be stated in the form

$$u_t = f(x, u^J), \quad x \in \Omega \subseteq \mathbb{R}^d, \quad t \in [0,T], \quad u \in \mathcal{B} \subseteq L^2. \tag{1.2.1}$$

Here, we use the notation u^J to denote that f is dependent on u and the partial derivatives of u with respect to the spatial coordinates $x_1, ..., x_n$ up to some order J, and we do not specify the space $\mathcal{B}$ since this will typically vary depending on the PDE and the boundary values imposed. PDEs of this kind are typically solved numerically using finite difference schemes or finite element schemes. Common to these schemes is that they require that (1.2.1) be *spatially discretized*, that is, the domain Ω is replaced with Ω_h, a discrete domain defined by a *mesh* $x^h = \{x_l\}_{l=1}^n \subseteq \Omega$. This has the effect of reducing problem (1.2.1) to a finite-dimensional problem

$$\dot{u}^h = f^h(x^h, u^h), \tag{1.2.2}$$

where $u^h(t) = (u_1^h(t), ..., u_m^h(t)) \in \mathbb{R}^m$ contains the spatially discrete degrees of freedom $u_l^h(t)$. The significance of the degrees of freedom depends on the method used. In standard finite difference methods, $u_l^h(t)$ is the approximate value of $u(x_l, t)$, and likewise for certain choices of basis in finite element methods, such as the standard polynomial bases, although this is not the case when using other bases such as spectral bases or B-spline bases [19] [6].

When spatially discretizing the PDE it is desirable that any existing structure of the PDE is inherited in the discretization (1.2.2) in a discrete sense. One such type of structure is first integral preservation. A function $I : \mathcal{B} \to \mathbb{R}$ is said to be a first integral of (1.2.1) if

$$\frac{\mathrm{d}}{\mathrm{d}t} I(u(t)) = \left\langle \frac{\delta I}{\delta u}, u_t \right\rangle = \left\langle \frac{\delta I}{\delta u}, f(x, u^J), \right\rangle = 0,$$

for all $u \in \mathcal{B}$ solving (1.2.1). Here, $\delta I / \delta u$ denotes the variational derivative of I with respect to u. A good discretization of the form (1.2.2) would have a corresponding first integral I^h that approximates I. In light of the previous discussion, it is then preferable to employ a conservative scheme for time stepping the spatially discrete equation. This is the approach used in the first three papers.

Producing conservative PDE solvers is the goal of the discrete variational derivative (DVD) method, developed in a number of papers [9–11, 29, 30] and culminating in the book [12]. The DVD method is a sophisticated system where the first integral is discretized before the machinery of discrete variation is used to obtain a numerical scheme that preserves the discrete energy. Discrete variation involves summation by parts, a direct analogue to the integration by parts used to prove first integral preservation in the continuous system. Note that it is also possible to obtain conservative solvers by first discretizing in time using e.g. discrete gradients, then in space as in [7]. It is also possible to reduce the computational burden of implicitness by using linearly implicit schemes as in [7] and [31].

For any spatial discretization the choice of mesh is of high importance since the accuracy of the numerical solution is often directly linked to the distance between points on the grid; finer grids correspond to more accurate solutions. This is especially the case for problems where difficult domains can cause instabilities and for PDEs where the interesting behaviour of the solution is localized, meaning the grid needs to be finer only in certain areas. The latter type of PDEs includes those with soliton solutions, travelling waves that do not change shape, such as the Sine-Gordon, Korteweg-de Vries (KdV) and Benjamin-Bona-Mahoney (BBM) equations [25,41,42]. Due to the importance of mesh choice, there has been much interest in adaptive strategies for tailoring the meshes to the problem at hand, resulting in three main modes of mesh adaptivity:

- r-adaptivity: Especially used for time-adaptive equations, one keeps the number of mesh points constant and moves them according to some rule, e.g. to follow wave fronts. This typically does not affect the number of degrees of freedom in the discretization.
- h-adaptivity: A general strategy based on inserting new mesh points and/or removing old mesh points to improve resolution in critical areas. This has the effect of increasing the number of degrees of freedom.
- p-adaptivity: Specifically for finite element-type methods, this approach improves accuracy by increasing the order of the basis function polynomials or, more generally, by improving the choice basis functions. Similar to h-adaptivity, this increases the number of degrees of freedom.

While adaptivity can dramatically improve the performance of a PDE solver, it is also possible that it influences the stability properties of the solver negatively. Hence, stabilizing properties such as energy preservation can be useful for improving adaptive schemes, in particular for nonlinear problems. This is, for instance, seen in [33], where a wavelet-based mesh adaptivity procedure is successfully combined with the DVD method to solve the KdV and Cahn-Hilliard equations. The first three papers of this thesis concern the amalgamation of spatially adaptive methods for PDEs and first integral preserving methods, where the temporal discretization is based on discrete gradient methods.

1.3 Optimization theory and image analysis

In the fourth and fifth papers we consider discrete gradient schemes as optimization methods, with applications in image analysis. This section is meant as a selective introduction to the concepts used in the papers, for more in-depth sources on these matters, see e.g. [40] or [38].

The goal of unconstrained optimization is to find a minimizer of an *objective function* $V : \mathbb{R}^n \to \mathbb{R}$, i.e. find a point $x^* \in \mathbb{R}^n$ such that $V(x^*) \leq V(x)$ for all $x \in \mathbb{R}^n$. The problem of minimizing V is well-defined if V is lower semi-continuous and coercive, where V is called lower semi-continuous if, given any $x \in \mathbb{R}^n$, then for all sequences $\{x^k\}_{k\in\mathbb{N}}$ converging to x,

$$f(x) \leq \liminf_{k\to\infty} f(x^k).$$

The objective function V is coercive if given a sequence $\{x^k\}_{k\in\mathbb{N}} \subseteq \mathbb{R}^n$ such that $\lim_{k\to\infty} \|x^k\| = \infty$, then $\lim_{k\to\infty} V(x^k) = \infty$. Being well-defined does not necessarily mean that the problem is easy to solve; the problem may be discontinuous or possess many local minimizers, making it hard to find a global optimum algorithmically.

There is a close relation between numerical integration and numerical optimization, stemming from the concept of gradient flows. If V is differentiable, the typical approach for using numerical integrators to minimize V is by numerically solving the gradient flow ODE

$$\dot{u}(t) = -\nabla V(u(t)) \tag{1.3.1}$$

from a chosen starting point u_0, until an approximate equilibrium is reached. Equation (1.3.1) is dissipative since

$$\frac{\mathrm{d}}{\mathrm{d}t} V(u(t)) = -\|\nabla V(u(t))\|^2 \leq 0,$$

where $\|\cdot\|$ denotes the Euclidian norm. Thus, if V is bounded from below, one may expect $u(t)$ to converge to a minimizer of V since the energy is constantly decreasing. An example of an optimization scheme based on numerical integration is the well-known gradient descent algorithm. This algorithm in its basic form is equvalent to applying a forward Euler scheme with possibly varying time steps τ_k for the time-discretization of (1.3.1) to get

$$u^{k+1} = u^k - \tau_k \nabla V(u^k).$$

To work well as an optimization algorithm, it is desirable that a numerical integrator is dissipative in the discrete sense: $V(u^{k+1}) - V(u^k) \leq 0$. As one might expect, a Runge-Kutta integrator is dissipative if the τ_k are chosen small enough, as seen in [20], but it will in general not be dissipative for all time steps. This is one of the challenges of using numerical integration schemes for optimization; the τ_k must be small to have stability and dissipation, but at the same time one wishes to take τ_k as large as possible such that a stationary point is reached quickly. Solving equation (1.3.1) exactly is of secondary importance. Therefore, to use an ODE scheme for numerical optimization, one must pay

special attention to stability and dissipation properties and how they relate to the choice of τ_k. An interesting development is the notion of using geometric numerical integration schemes for this purpose, an idea first considered in [15] where discrete gradient schemes are used for large-scale problems in image analysis, in part due to their dissipation property which holds regardless of τ_k. Also, one can consider the successive over-relaxation (SOR) method for linear systems as the Itoh–Abe discrete gradient applied to a the gradient flow of a quadratic objective function [34].

Certain optimization problems are easier to solve than others, stemming from properties of the objective function. One property in particular that optimization easier is convexity. A function $V : \mathbb{R}^n \to \mathbb{R}$ is said to be convex if

$$V(tx + (1-t)y) \leq tV(x) + (1-t)V(y) \quad \text{for all} \quad t \in (0,1)$$

and if it is differentiable, it is said to be strongly convex with parameter $\sigma > 0$ if

$$V(y) \leq V(x) + \langle \nabla V(x), y - x \rangle + \frac{\sigma}{2}\|y - x\|^2.$$

In the case when V is L-smooth, i.e. differentiable with Lipschitz continuous gradient, one can often obtain similar results by considering functions satisfying the Polyak-Łojasiewicz (PŁ) inequality

$$\frac{1}{2}\|\nabla V(x)\|^2 \geq \sigma(f(x) - f^*)$$

for some $\sigma > 0$. These functions, which need not be convex, are called PŁ-functions, and this property is the assumption under which Polyak shows linear convergence of gradient descent methods in the [43]. The class of L-smooth, strongly convex functions is shown to be included in the class of PŁ-functions in [23].

One can also make sense of strong convexity for nonsmooth V, but this is outside the scope of the later articles and hence not presented here. A convex objective function has many additional properties that can be exploited in the construction of numerical optimizers. For example, any minimizer of a convex function is a global minimizer, and in the case of strong convexity it is unique, even for non-smooth problems. One also has a well-developed duality theory for convex problems, which we will not expand upon here but mention that it is useful for constructing efficient optimization algorithms.

These efficient algorithms for convex problems include methods like the interior-point methods [39] and primal-dual methods like the one presented in [5], that work even if the objective function is nonsmooth. It is also worth noting that convex functions are easier to minimize in the sense that convergence rates

for standard methods are faster for convex functions than non-convex functions. For example, the gradient descent algorithm has $\mathcal{O}(1/k)$ convergence, i.e.

$$V(u^k) - V^* \leq \frac{C_1}{k}$$

for convex problems and linear convergence, i.e.

$$V(u^k) - V^* \leq C_2^k (V(u^0) - V^*),$$

with $C_2 \in (0,1)$, for strictly convex problems [38] or when V is a PŁ-function [23]. In the fourth paper we consider the use of a discrete gradient based optimization scheme to solve a non-convex optimization problem from digital image processing and in the fifth paper we consider the theoretical underpinnings of discrete gradient based optimization, showing convergence rates when the objective function is convex or a PŁ-function.

1.3.1 Variational image analysis

Digital image processing encompasses many tasks for the improvement and analysis of images. These tasks include, for example, image restoration, segmentation, classification and motion tracking. There are several ways of solving these problems such as filtering, stochastic methods and machine learning. Another popular method is through variational image analysis, where image analysis problems are cast as optimization problems; a typical form of this is to consider an image as a function $u : \Omega \to \mathbb{R}$ and to minimize an objective function $V : \mathbb{R}^n \to \mathbb{R}$ given by

$$V(u) = d(Ku, g) + \alpha J(u) \tag{1.3.2}$$

where $g : \mathbb{R}^n \to \mathbb{R}$ is the damaged input image, d is a distance function, K a forward operator relating the output image u to g (i.e. by blurring u if g is blurred), J is a regularization function and $\alpha \in \mathbb{R}$ is the regularization strength. Differences between u and g are enforced by the regularizer J, which acts to punish unwanted behaviour in u. An example, popular in image denoising problems, is the Rudin-Osher-Fatemi (ROF) regularizer where $J(u)$ is a measure of the total variation of u, with the idea being that noise causes higher variation in a natural image [45], and that by lowering variation in g one obtains a less noisy output image u. The choice of $\mathbb{R}$ as the image of u corresponds to greyscale image processing, since u assigns a single real-valued number to each point, which can be interpreted as a greyscale intensity. Choosing $\mathrm{Im}(V) = \mathbb{R}^3$ is typical of colour images such as RGB images, and yet other choices such as $\mathrm{Im}(V) = S^1 \times \mathbb{R}^2$, where S^1 is the unit circle, exist. Image processing with manifold-valued images is one of the examples considered in the sixth paper.

There are several methods of minimizing (1.3.2), one of which is the PDE-based approach where one considers its Euler-Lagrange equations, exemplified in the lagged-diffusivity method of solving the ROF regularization problem [47]. Another method, based on the discrete nature of pixel format images is to discretize (1.3.2) using difference methods and quadrature, then minimizing the discretized functional, an approach that is similar to the one used for spatially discretizing PDEs in the preceding section. The latter method is often the most practical, provided that the solutions converge to the continuous solution for finer discretizations. The discretization approach is used in the case of Euler's elastica regularization considered in the fourth paper, where we employ an optimization algorithm based on the Itoh–Abe discrete gradient.

1.4 Generalizations to manifolds

The following discussion is meant to provide background on the Riemannian manifolds considered in papers 6 and 7. It is based on the books of Lang and Lee [24, 26], and the interested reader is advised to consult these books for more details.

Some problems in optimization and ODE theory are, due to their structure, most naturally formulated on a finite-dimensional manifold M. Simple examples of such problems include the ROF denoising problem in HSV space mentioned above, where one component is located on the circle S^1, and certain problems in computational mechanics where it is natural to express orientations of objects by vectors on the sphere S^2. Furthermore, there is a great variety of problems that are best formulated on Lie groups.

A fundamental question when considering computational problems on manifolds is representation of the manifold. For example, if $M = S^2$, one can either take the extrinsic view of considering the sphere as an embedded manifold of $\mathbb{R}^3$, or one can take the intrinsic view and represent it by orthogonal rotations in $\mathrm{SO}(3)$. The intrinsic view has had great success, for example with the Runge-Kutta-Munthe-Kaas methods for ODEs on Lie groups developed in [21, 35–37]. When working in the intrinsic view, it is necessary to make generalizations of concepts from calculus in Euclidean spaces. In the following, we will consider finite-dimensional Riemannian manifolds since they possess all the necessary structure needed to make sense of gradients, which are essential to the formulation of dissipative systems in the sixth paper, and to the extension of the Itoh–Abe discrete gradient to the manifold setting.

A smooth, finite-dimensional manifold M is a space that is locally homeomorphic to $\mathbb{R}^n$; n is said to be its dimension. It has a well-defined differentiable structure based on tangent vectors; for each point $p \in M$, there is an associated tangent space T_pM consisting of the equivalence class of curves $\gamma : \mathbb{R} \to M$

such that $\gamma(0) = p$, with equivalence between curves γ and λ being defined by $\gamma \equiv \lambda$ if $\dot{\gamma}(0) = \dot{\lambda}(0)$. The tangent space T_pM is a vector space so that notions of scaling and addition, which are not naturally defined on M, exist on T_pM. The disjoint union of all tangent spaces is called the tangent bundle TM:

$$TM = \bigcup_{p \in M} \left\{ (p, v) | v \in T_pM \right\}.$$

There is a canonical projection operator $\pi : TM \to M$ that works with disjoint structure of TM, given by $\pi((p, v)) = p$. This allows us to make sense of vector fields on M; a vector field X on M is a section of TM, i.e. a continuous function $X : M \to TM$ such that $\pi(X(p)) = p$. With this notion at hand, we can consider differential equations on M,

$$\dot{x}(t) = F(x), \qquad x(0) = x_0 \in M, \tag{1.4.1}$$

with F being a vector field. As with ODEs in Euclidean spaces it is often necessary to solve these numerically. Of particular interest in numerical optimization is the gradient flow, which is well-defined on Riemannian manifolds. A Riemannian manifold is a smooth manifold equipped with a Riemannian metric g - a symmetric, positive definite (0,2)-tensor field. The Riemannian metric introduces a smoothly varying inner product on the tangent spaces T_pM, allowing us to naturally define gradient vector fields.

A typical numerical method, be it for solving ODEs or for optimization, requires the notion of moving in a certain direction. In Euclidean spaces, this is achieved by following straight lines by vector addition, while on smooth manifolds the notion of a straight line is generalized by introducing an affine connection which in turn can be used to define *geodesics*. The Riemannian metric g induces a natural choice of affine connection on M, called the Levi-Civita connection, and also a metric function $d : M \times M \to \mathbb{R}$ on M. All geodesics with respect to the Levi-Civita connection are locally length-minimizing, which makes them natural extensions of real lines to the manifold setting.

On a Riemannian manifold, the notion of moving along a geodesic is encoded in the Riemannian exponential map $\exp : TM \to M$, which maps $(p, v) \in TM$ to the point $\exp_p(v) \in M$, obtained by following the geodesic passing through p with direction v until $t = 1$. It is similar to the exponential map of Lie group theory that is central to the Runge-Kutta-Munthe-Kaas methods. In fact, if G is a compact Lie group one can define a left and right translationally invariant Riemannian metric g on it, and the Lie group exponential on G coincides with the Riemannian exponential with respect to g . Several ODE solvers and optimization methods based on the Riemannian exponential exist in the literature [27] [46]. A drawback is that the exponential may not exist in closed form, and even if it does, it may be too expensive to compute with

sufficient accuracy in many applications, e.g. if it relies on matrix exponentiation. This drawback can be reduced by considering alternative tangent space parametrizations called retractions, used for solving conservative ODEs in [4] and for optimization in [1] [3]. The methods presented in papers 6 and 7 concern dissipative and conservative ODEs on Riemannian manifolds, respectively, using retraction mappings.

1.5 Summary of papers

The following papers constitute the scientific contribution of this thesis. To fit the thesis format, the layout and typography of the papers has been altered, as well as certain sentences and equations, but not beyond what should be characterised as cosmetic. No alterations to the scientific content of the papers have been made.

PAPER 1: Adaptive first integral preserving methods for partial differential equations

Torbjørn Ringholm
Published in: *Proceedings in Applied Mathematics and Mechanics 16 (2016)*

This conference proceedings paper is a prelude to paper 2 and presents an abridged version of the contents of the latter. A method of spatially discretizing conservative PDEs of a certain form based on a discretized energy is shown. The spatially discretized system conserves the discretized energy. The energy discretization is done using quadrature on an arbitrary fixed grid, resulting in a general finite difference discretization. Then, using discrete gradients for time stepping and a projection step, we show how to preserve the discretized energy when the grid is adaptive. Numerical results are shown for the Sine-Gordon equation using a finite difference discretization and an r-adaptive moving mesh strategy.

PAPER 2: Adaptive energy preserving methods for partial differential equations

Sølve Eidnes, Brynjulf Owren and Torbjørn Ringholm
Published in: *Advances in Computational Mathematics 44 (3), 815-839 (2018)*

This is the main article that the conference proceedings papers 1 and 3 are based on. It presents a framework for generating conservative schemes for PDEs presented in a skew-symmetric form, using both finite differences and the partition of unity method for spatial discretization and discrete gradient

schemes for time stepping. We show an equivalence between a certain type of DVD methods and the discrete gradient systems schemes obtained. Furthermore, we present an idea for combining the conservative methods with spatial adaptivity, then discuss how to map solutions between grids at different step sizes in both preserving and non-preserving manners and how results obtained by projection methods can be considered a subclass of the discrete gradient methods in the adaptive setting. The numerical experiments presented concern soliton solutions of the Sine-Gordon equation and the KdV equation. The Sine-Gordon equation is discretized with standard finite elements and uses a more sophisticated interpolation/adaptivity method than that considered in paper 1, while the KdV equation is discretized using the finite element method.

PAPER 3: Energy preserving moving mesh methods applied to the BBM equation

Sølve Eidnes and Torbjørn Ringholm
Published in: *Proceedings of MekIT '17 (2017)*

This conference proceedings paper is based on paper 2. It concerns energy preserving integration of spatially adaptive discretizations of conservative PDEs. We apply the methods to the numerical computation of soliton solutions of the Benjamin-Bona-Mahoney equation, using an r-adaptive finite elements approach with third-order B-spline basis functions for the spatial discretization, and the AVF discrete gradient for time stepping. The BBM equation has three conservation laws; we consider two schemes that preserve different energies and compare the results.

PAPER 4: Variational image analysis with Euler's elastica using a discrete gradient scheme

Torbjørn Ringholm, Jasmina Lazić and Carola-Bibiane Schönlieb
Submitted

This paper marks a change in direction from the PDE-based works of papers 1-3. Here, discrete gradient methods, and in particular the Itoh–Abe discrete gradient, are used for numerical optimization. The Itoh–Abe discrete gradient applied to the gradient system associated with an energy is considered as an alternative to gradient descent and coordinate descent methods, and its convergence is analysed for non-convex problems. A result shows that the number of iterations needed for convergence does not depend on the problem size in certain cases. A strategy for parallelization of the algorithm is presented, and

numerical results show its use in variational image analysis problems using Euler's elastica as a regularizing prior. Other numerical tests concern convergence rates and execution time, compared to other standard algorithms.

PAPER 5: A geometric integration approach to smooth optimisation: Foundations of the discrete gradient method

Matthias Ehrhardt, Erlend Riis, Torbjørn Ringholm and Carola-Bibiane Schönlieb
To be submitted

In this paper, we address several fundamental questions about the discrete gradient method when used as an optimization scheme. We provide existence results of the solutions to the implicit systems that define the iterates for various discrete gradients for all time steps, under certain assumptions on the objective function. We also analyse the dependence of the iterates on the choice of time step. Finally, we establish convergence rates for convex functions with Lipschitz continuous gradients, and PŁ-functions. We consider three types of discrete gradients: the Gonzalez discrete gradient, the mean value discrete gradient and the Itoh–Abe discrete gradient. We also consider a randomised version of the Itoh–Abe discrete gradient based optimization method.

PAPER 6: Dissipative numerical schemes on Riemannian manifolds with applications to gradient flows

Elena Celledoni, Sølve Eidnes, Brynjulf Owren and Torbjørn Ringholm
Submitted

This paper concerns the use of discrete Riemannian gradients for dissipative problems on Riemannian manifolds. Building on the work in [4], we introduce discrete Riemannian gradients, generalizing discrete gradient methods to Riemannian manifolds. We also introduce the Itoh–Abe discrete gradient in this setting, and apply it to gradient flow systems to obtain an optimization scheme. This scheme's convergence is analysed. We apply the method to eigenvalue problems and to denoising problems in interferometric synthetic aperture radar and diffusion tensor imaging.

PAPER 7: Energy preserving methods on Riemannian manifolds

Elena Celledoni, Sølve Eidnes, Brynjulf Owren and Torbjørn Ringholm
Submitted

The final paper is related to the sixth paper in considering discrete Riemannian gradient (DRG) methods for ODEs on Riemannian manifolds. We formulate the AVF and Midpoint DRGs using a specialization of the discrete differentials of [4] and present a convergence order result concerning all DRGs. We then extend the DRGs to higher order methods using an energy-preserving collocation-like method as in [16] and, in the case of the Itoh–Abe discrete gradient, symmetrization. We apply the resulting schemes to two variants of spin systems from physics and verify the convergence order of the methods.

Bibliography

[1] P.-A. ABSIL, R. MAHONY, AND R. SEPULCHRE, *Optimization algorithms on matrix manifolds*, Princeton University Press, 2009.

[2] M. P. CALVO, A. ISERLES, AND A. ZANNA, *Numerical solution of isospectral flows*, Math. Comp., 66 (1997), pp. 1461–1486.

[3] E. CELLEDONI AND S. FIORI, *Descent methods for optimization on homogeneous manifolds*, Math. Comput. Simulation, 79 (2008), pp. 1298–1323.

[4] E. CELLEDONI AND B. OWREN, *Preserving first integrals with symmetric Lie group methods*, Discrete Contin. Dyn. Syst., 34 (2014), pp. 977–990.

[5] A. CHAMBOLLE AND T. POCK, *A first-order primal-dual algorithm for convex problems with applications to imaging*, J. Math. Imaging Vis., 40 (2011), pp. 120–145.

[6] J. A. COTTRELL, T. J. R. HUGHES, AND Y. BAZILEVS, *Isogeometric analysis*, John Wiley & Sons, Ltd., Chichester, 2009.

[7] M. DAHLBY AND B. OWREN, *A general framework for deriving integral preserving numerical methods for PDEs*, SIAM J. Sci. Comput., 33 (2011), pp. 2318–2340.

[8] M. DAHLBY, B. OWREN, AND T. YAGUCHI, *Preserving multiple first integrals by discrete gradients*, J. Phys. A, 44 (2011).

[9] D. FURIHATA, *Finite difference schemes for $\partial u/\partial t = (\partial/\partial x)^{\alpha}\delta G/\delta u$ that inherit energy conservation or dissipation property*, J. Comput. Phys., 156 (1999), pp. 181–205.

[10] D. FURIHATA, *A stable and conservative finite difference scheme for the Cahn-Hilliard equation*, Numer. Math., 87 (2001), pp. 675–699.

[11] D. FURIHATA AND T. MATSUO, *A stable, convergent, conservative and linear finite difference scheme for the Cahn-Hilliard equation*, Japan J. Indust. Appl. Math., 20 (2003), pp. 65–85.

[12] D. FURIHATA AND T. MATSUO, *Discrete variational derivative method*, Chapman & Hall/CRC Numerical Analysis and Scientific Computing, CRC Press, Boca Raton, FL, 2011.

[13] H. H. GOLDSTINE, *A history of numerical analysis from the 16th through the 19th century*, Springer-Verlag, New York-Heidelberg, 1977.

[14] O. GONZALEZ, *Time integration and discrete Hamiltonian systems*, J. Nonlinear Sci., 6 (1996), pp. 449–467.

[15] V. GRIMM, R. I. MCLACHLAN, D. I. MCLAREN, G. QUISPEL, AND C. SCHÖNLIEB, *Discrete gradient methods for solving variational image regularisation models*, J. Phys. A: Math. Theor., 50 (2017).

[16] E. HAIRER, *Energy-preserving variant of collocation methods*, J. Numer. Anal. Ind. Appl. Math., 5 (2010), pp. 73–84.

[17] E. HAIRER, C. LUBICH, AND G. WANNER, *Geometric numerical integration*, vol. 31 of Springer Series in Computational Mathematics, Springer, Heidelberg, 2010.

[18] A. HARTEN, P. D. LAX, AND B. VAN LEER, *On upstream differencing and Godunov-type schemes for hyperbolic conservation laws*, SIAM Rev., 25 (1983), pp. 35–61.

[19] J. S. HESTHAVEN, S. GOTTLIEB, AND D. GOTTLIEB, *Spectral methods for time-dependent problems*, vol. 21 of Cambridge Monographs on Applied and Computational Mathematics, Cambridge University Press, Cambridge, 2007.

[20] A. HUMPHRIES AND A. STUART, *Runge–Kutta methods for dissipative and gradient dynamical systems*, SIAM J. Numer. Anal., 31 (1994), pp. 1452–1485.

[21] A. ISERLES, H. Z. MUNTHE-KAAS, S. P. NØRSETT, AND A. ZANNA, *Lie-group methods*, Acta Numer., 9 (2000), pp. 215–365.

[22] T. ITOH AND K. ABE, *Hamiltonian-conserving discrete canonical equations based on variational difference quotients*, J. Comput. Phys., 76 (1988), pp. 85–102.

[23] H. KARIMI, J. NUTINI, AND M. SCHMIDT, *Linear convergence of gradient and proximal-gradient methods under the Polyak–Łojasiewicz condition*, in ECML PKDD, Springer, 2016, pp. 795–811.

[24] S. LANG, *Fundamentals of differential geometry*, vol. 191, Springer Science & Business Media, 2012.

[25] P. D. LAX, *Integrals of nonlinear equations of evolution and solitary waves*, Comm. Pure Appl. Math., 21 (1968), pp. 467–490.

[26] J. M. LEE, *Riemannian manifolds: an introduction to curvature*, vol. 176, Springer Science & Business Media, 2006.

[27] B. LEIMKUHLER AND G. W. PATRICK, *A symplectic integrator for Riemannian manifolds*, J. Nonlinear Sci., 6 (1996), pp. 367–384.

[28] B. LEIMKUHLER AND S. REICH, *Simulating Hamiltonian dynamics*, vol. 14 of Cambridge Monographs on Applied and Computational Mathematics, Cambridge University Press, Cambridge, 2004.

[29] T. MATSUO, *New conservative schemes with discrete variational derivatives for nonlinear wave equations*, J. Comput. Appl. Math., 203 (2007), pp. 32–56.

[30] T. MATSUO, *Dissipative/conservative Galerkin method using discrete partial derivatives for nonlinear evolution equations*, J. Comput. Appl. Math., 218 (2008), pp. 506–521.

[31] T. MATSUO AND D. FURIHATA, *Dissipative or conservative finite-difference schemes for complex-valued nonlinear partial differential equations*, J. Comput. Phys., 171 (2001), pp. 425–447.

[32] R. I. MCLACHLAN, G. R. W. QUISPEL, AND N. ROBIDOUX, *Geometric integration using discrete gradients*, Philos. T. R. Soc. A, 357 (1999), pp. 1021–1045.

[33] Y. MIYATAKE AND T. MATSUO, *A note on the adaptive conservative/dissipative discretization for evolutionary partial differential equations*, J. Comput. Appl. Math., 274 (2015), pp. 79–87.

[34] Y. MIYATAKE, T. SOGABE, AND S.-L. ZHANG, *On the equivalence between SOR-type methods for linear systems and discrete gradient methods for gradient systems*, arXiv preprint arXiv:1711.02277, (2017).

[35] H. MUNTHE-KAAS, *Lie-Butcher theory for Runge-Kutta methods*, BIT, 35 (1995), pp. 572–587.

[36] H. MUNTHE-KAAS, *High order Runge-Kutta methods on manifolds*, in Proceedings of the NSF/CBMS Regional Conference on Numerical Analysis of Hamiltonian Differential Equations (Golden, CO, 1997), vol. 29, 1999, pp. 115–127.

[37] H. MUNTHE-KAAS AND A. ZANNA, *Numerical integration of differential equations on homogeneous manifolds*, in Foundations of Computational Mathematics (Rio de Janeiro, 1997), Springer, Berlin, 1997, pp. 305–315.

[38] Y. NESTEROV, *Introductory lectures on convex optimization: A basic course*, vol. 87, Springer Science & Business Media, 2013.

[39] Y. NESTEROV AND A. NEMIROVSKII, *Interior-point polynomial algorithms in convex programming*, vol. 13, SIAM, 1994.

[40] J. NOCEDAL AND S. WRIGHT, *Numerical Optimization*, Springer Series in Operations Research and Financial Engineering, Springer-Verlag, New York, 2 ed., 2006.

[41] P. J. OLVER, *Euler operators and conservation laws of the BBM equation*, Math. Proc. Cambridge Philos. Soc., 85 (1979), pp. 143–160.

[42] J. K. PERRING AND T. H. R. SKYRME, *A model unified field equation*, Nuclear Phys., 31 (1962), pp. 550–555.

[43] B. T. POLJAK, *Gradient methods for minimizing functionals*, Ž. Vyčisl. Mat. i Mat. Fiz., 3 (1963), pp. 643–653.

[44] A. QUARTERONI, R. SACCO, AND F. SALERI, *Numerical mathematics*, vol. 37, Springer Science & Business Media, 2010.

[45] L. I. RUDIN, S. OSHER, AND E. FATEMI, *Nonlinear total variation based noise removal algorithms*, Physica D, 60 (1992), pp. 259–268.

[46] C. UDRISTE, *Convex functions and optimization methods on Riemannian manifolds*, vol. 297, Springer Science & Business Media, 1994.

[47] C. R. VOGEL AND M. E. OMAN, *Iterative methods for total variation denoising*, SIAM J. Sci. Comput., 17 (1996), pp. 227–238.

[48] H. XUE AND A. ZANNA, *Explicit volume-preserving splitting methods for polynomial divergence-free vector fields*, BIT, 53 (2013), pp. 265–281.

Adaptive first integral preserving methods for partial differential equations

Torbjørn Ringholm

Published in Proceedings in Applied Mathematics and Mechanics 16, pp. 945–948 (2016)

Adaptive first integral preserving methods for partial differential equations

Abstract. We present a method for constructing first integral preserving numerical schemes for time-dependent partial differential equations on non-uniform grids, using a finite difference approach for spatial discretization and discrete gradients for time stepping. The method is extended to accommodate spatial adaptivity. A numerical experiment is carried out where the method is applied to the Sine-Gordon equation with moving mesh adaptivity.

2.1 Introduction

Following the success of geometric integrators for ordinary differential equations (ODEs) [8], there has been an interest in developing similar techniques for numerical integration of partial differential equations (PDEs) conserving properties such as multi-symplecticity [2]. A feature of many PDEs is the conservation of energy-like quantities, generally called first integrals. During the last two decades, there have been developed schemes for PDEs that conserve first integrals numerically, such as the discrete variational derivative (DVD) schemes of Furihata, Matsuo, Sugihara and Yaguchi [6], and schemes using tools from the ODE literature [3, 4]. Most of these schemes use finite differences on uniform grids for spatial discretization, with some notable exceptions.

Yaguchi, Matsuo and Sugihara present in [12] and [13] two DVD schemes on fixed, non-uniform grids. The use of non-uniform grids is important for multidimensional problems, since using uniform grids restricts the types of domains possible to discretize. Another reason for using non-uniform grids is that it allows for mesh adaptivity [1, 14]. Adaptive energy preserving schemes for the Korteweg-de Vries and Cahn-Hilliard equations have been developed recently by Miyatake and Matsuo [11]. We shall propose a general framework for PDE solvers combining adaptivity with first integral conservation. We have based our methods on discrete gradient methods for ODEs, often attributed to Gonzalez [7]. Presented here is an abridged version of our work, using finite differences for spatial discretization and numerical experiments on the Sine-Gordon equation. A full version including the use of variational methods and an application to the KdV equation, can be found in [5].

2.2 Problem statement and fixed grid discretization

Consider a PDE, where u^J denotes dependence on u and its partial derivatives with respect to $x_1, ..., x_d$:

$$u_t = f(\mathbf{x}, u^J), \qquad \mathbf{x} \in \Omega \subseteq \mathbb{R}^d, \quad u \in \mathcal{B} \subseteq L^2, \tag{2.2.1}$$

where we assume $\mathcal{B}$ is sufficiently regular to allow all following operations. A first integral of (2.2.1) is a functional $\mathcal{I}[u]$ satisfying

$$\left\langle \frac{\delta \mathcal{I}}{\delta u}[u], f(\mathbf{x}, u^J) \right\rangle_{L^2} = 0, \quad \forall u \in \mathcal{B},$$

where $\frac{\delta \mathcal{I}}{\delta u}$ is the variational derivative of $\mathcal{I}$. Observe that $\mathcal{I}[u]$ is conserved since $\frac{\mathrm{d}\mathcal{I}}{\mathrm{d}t} = 0$. If there exists a skew-symmetric operator $S(\mathbf{x}, u^J)$ such that $f(\mathbf{x}, u^J) = S(\mathbf{x}, u^J)\frac{\delta \mathcal{I}}{\delta u}[u]$, then $\mathcal{I}[u]$ is a first integral of (2.2.1), and we can state (2.2.1) as

$$u_t = S(\mathbf{x}, u^J)\frac{\delta \mathcal{I}}{\delta u}[u]. \tag{2.2.2}$$

This can be considered as the PDE analogue of an ODE with a first integral, where we have a system

$$\dot{\mathbf{y}} = S(\mathbf{y})\nabla I(\mathbf{y}), \tag{2.2.3}$$

with $S(\mathbf{y})$ a skew-symmetric matrix [10]. Note that Hamiltonian ODEs are contained in this class of ODEs. For ODEs of the form (2.2.3), first integral preserving numerical methods exist, i.e. discrete gradient methods:

$$\mathbf{y}^{n+1} - \mathbf{y}^n = \Delta t \bar{S}(\mathbf{y}^n, \mathbf{y}^{n+1})\overline{\nabla} I(\mathbf{y}^n, \mathbf{y}^{n+1}),$$

where $\bar{S}(\mathbf{y}^n, \mathbf{y}^{n+1})$ is a skew-symmetric approximation to $S(\mathbf{y})$ and $\overline{\nabla} I(\mathbf{v}, \mathbf{w})$ a discrete gradient of $I(\mathbf{y})$, satisfying

$$(\overline{\nabla} I(\mathbf{v}, \mathbf{w}))^T (\mathbf{w} - \mathbf{v}) = I(\mathbf{w}) - I(\mathbf{v}), \qquad \overline{\nabla} I(\mathbf{w}, \mathbf{w}) = \nabla I(\mathbf{w}).$$

There are several choices of discrete gradients available, we will use the Average Vector Field (AVF) discrete gradient [3]:

$$\overline{\nabla} I(\mathbf{v}, \mathbf{w}) = \int_0^1 \nabla I(\xi \mathbf{w} + (1-\xi)\mathbf{v}) \mathrm{d}\xi.$$

2.2.1 Finite difference method on fixed grid

Given grid points $\mathbf{x}_0, ..., \mathbf{x}_N$, we interpret them as quadrature points with quadrature weights $\kappa_0, ..., \kappa_N$ and approximate the L^2 inner product by a weighted inner product:

$$\langle u, v \rangle_{L^2} = \int_\Omega u(\mathbf{x}) v(\mathbf{x}) \mathrm{d}x \simeq \sum_{i=0}^{N} \kappa_i u(\mathbf{x}_i) v(\mathbf{x}_i) = \mathbf{u}^T D(\kappa) \mathbf{v} = \langle \mathbf{u}, \mathbf{v} \rangle_\kappa,$$

where $D(\kappa) = \text{diag}(\kappa_0, ..., \kappa_N)$. Next, assuming that we have a consistent mesh dependent approximation $\mathcal{I}_{\mathbf{x}}(\mathbf{u})$ to $\mathcal{I}[u]$, we discretize $\frac{\delta \mathcal{I}}{\delta u}$ by asserting that

$$\left\langle \frac{\delta \mathcal{I}_{\mathbf{x}}}{\delta \mathbf{u}}(\mathbf{u}), \mathbf{v} \right\rangle_{\kappa} = \left. \frac{\mathrm{d}}{\mathrm{d}\epsilon} \right|_{\epsilon=0} \mathcal{I}_{\mathbf{x}}(\mathbf{u} + \epsilon \mathbf{v}) \quad \forall \mathbf{v} \in \mathbb{R}^{N+1}$$

that is,

$$\left(\frac{\delta \mathcal{I}_{\mathbf{x}}}{\delta \mathbf{u}}(\mathbf{u}) \right)^T D(\kappa)\mathbf{v} = (\nabla \mathcal{I}_{\mathbf{x}}(\mathbf{u}))^T \mathbf{v} \quad \forall \mathbf{v} \in \mathbb{R}^{N+1},$$

from which we conclude that $\frac{\delta \mathcal{I}_{\mathbf{x}}}{\delta \mathbf{u}}(\mathbf{u}) = D(\kappa)^{-1} \nabla \mathcal{I}_{\mathbf{x}}(\mathbf{u})$. Using $\frac{\delta \mathcal{I}_{\mathbf{x}}}{\delta \mathbf{u}}$ as a discretization of $\frac{\delta \mathcal{I}}{\delta u}$ and approximating $S(\mathbf{x}, u^J)$ by a matrix $S_d(\mathbf{u})$, skew-symmetric with respect to $\langle \cdot, \cdot \rangle_{\kappa}$, we obtain a discretization of (2.2.2) as:

$$\dot{\mathbf{u}} = S_{\mathbf{x}}(\mathbf{u}) \nabla \mathcal{I}_{\mathbf{x}}(\mathbf{u}), \tag{2.2.4}$$

where $S_{\mathbf{x}}(\mathbf{u}) = S_d(\mathbf{u}) D(\kappa)^{-1}$. This ODE is of the form (2.2.3), so by using a discrete gradient $\overline{\nabla} \mathcal{I}_{\mathbf{x}}$, and a skew-symmetric, time-discrete approximation $S_{\mathbf{x}}(\mathbf{u}^n, \mathbf{u}^{n+1})$ to $S_{\mathbf{x}}(\mathbf{u})$, we obtain the following scheme for which $\mathcal{I}_{\mathbf{x}}(\mathbf{u}^{n+1}) = \mathcal{I}_{\mathbf{x}}(\mathbf{u}^n)$:

$$\mathbf{u}^{n+1} - \mathbf{u}^n = \Delta t S_{\mathbf{x}}(\mathbf{u}^n, \mathbf{u}^{n+1}) \overline{\nabla} \mathcal{I}_{\mathbf{x}}(\mathbf{u}^n, \mathbf{u}^{n+1}). \tag{2.2.5}$$

2.2.2 Adaptive discretization

Using adaptivity, we update the location of the grid points at each time step. Let $\mathbf{x}^n$ and $\mathbf{x}^{n+1}$ denote the collection of grid points at the previous and next time steps. We must transfer $\mathbf{u}^n$ to the new parameter set before advancing in time, for example by interpolating using splines. Let $\hat{\mathbf{u}}$ denote the values of $\mathbf{u}^n$ transferred onto $\mathbf{x}^{n+1}$. The transfer operation called preserving if $\mathcal{I}_{\mathbf{x}^{n+1}}(\hat{\mathbf{u}}) = \mathcal{I}_{\mathbf{x}^n}(\mathbf{u}^{\mathbf{n}})$. If the transfer is preserving, we may take the next time step with a preserving scheme, e.g.

$$\mathbf{u}^{n+1} - \hat{\mathbf{u}} = \Delta t S_{\mathbf{x}^{n+1}}(\mathbf{u}^{n+1}, \hat{\mathbf{u}}) \overline{\nabla} \mathcal{I}_{\mathbf{x}^{n+1}}(\mathbf{u}^{n+1}, \hat{\mathbf{u}}),$$

for which $\mathcal{I}_{\mathbf{x}^{n+1}}(\mathbf{u}^{n+1}) - \mathcal{I}_{\mathbf{x}^n}(\mathbf{u}^n) = 0$. If the transfer is non-preserving, as is the case with spline interpolation in general, we need to add a correction term to recover the preservation property. The scheme

$$\mathbf{u}^{n+1} - \hat{\mathbf{u}} = \Delta t S_{\mathbf{x}_{n+1}}(\mathbf{u}^{n+1}, \hat{\mathbf{u}}) \overline{\nabla} \mathcal{I}_{\mathbf{x}^{n+1}}(\mathbf{u}^{n+1}, \hat{\mathbf{u}}) - \frac{(\mathcal{I}_{\mathbf{x}^{n+1}}(\hat{\mathbf{u}}) - \mathcal{I}_{\mathbf{x}^n}(\mathbf{u}^n))\Phi}{\left\langle \overline{\nabla} \mathcal{I}_{\mathbf{x}^{n+1}}(\mathbf{u}^{n+1}, \hat{\mathbf{u}}), \Phi \right\rangle} \tag{2.2.6}$$

is first integral preserving in the sense that $\mathcal{I}_{\mathbf{x}^{n+1}}(\mathbf{u}^{n+1}) - \mathcal{I}_{\mathbf{x}^n}(\mathbf{u}^n) = 0$. The correcting direction Φ should be chosen so the correction is minimal and $\langle \overline{\nabla} \mathcal{I}_{\mathbf{x}^{n+1}}(\mathbf{u}^{n+1}, \hat{\mathbf{u}}), \Phi \rangle \neq 0$. We use $\Phi = \overline{\nabla} \mathcal{I}_{\mathbf{x}^{n+1}}(\mathbf{u}^{n+1}, \hat{\mathbf{u}})$ in the numerical experiment.

2.3 Numerical experiments

We applied our method to the Sine-Gordon Equation, a nonlinear hyperbolic PDE in 1+1 dimensions which has soliton solutions and therefore is a candidate for moving mesh adaptivity, where grid points are redistributed at each time step to cluster in areas of large gradients. Four methods were tested: a fixed mesh method with energy preservation by discrete gradients (DG) as in (2.2.5); a moving mesh method with preservation by discrete gradients (DGMM) as in (2.2.6); a non-preserving fixed grid method (MP), and a non-preserving moving mesh method (MPMM). The latter two were included to have a basis for comparison of our methods with standard methods. They are based on a finite difference scheme with spatial discretization by central finite differences and time discretization by the implicit midpoint rule. Cubic spline interpolation was used for transferring in the adaptive methods (DGMM and MPMM).

2.3.1 Adaptivity

Concerning adaptivity of the mesh, we used a simple moving mesh method based on an equidistribution principle which can be applied to problems in one spatial dimension. When $\Omega = [a, b]$ is split into N intervals $\{[x_i, x_{i+1}]\}_{i=0}^{i=N-1}$, one requires that the x_j be chosen such that

$$\int_{x_i}^{x_{i+1}} \sqrt{1+u_x^2}\,\mathrm{d}x = \frac{1}{N}\int_a^b \sqrt{1+u_x^2}\,\mathrm{d}x \quad \forall i.$$

That is, we require that the arc length of u is equidistributed over each interval. We only have an approximation of u, so a finite difference approximation was used to obtain approximately equidistributing grids with de Boor's method as in [9, pp. 36-38].

2.3.2 Sine-Gordon Equation

The Sine-Gordon Equation is stated in initial value problem form as:

$$u_{tt} - u_{xx} + \sin(u) = 0, \quad (x,t) \in \mathbb{R} \times [0,T], \quad u(x,0) = f(x), \quad u_t(x,0) = g(x). \tag{2.3.1}$$

We considered a finite domain $[-L, L] \times [0, T]$ with boundary conditions $u(-L) = u(L)$ and $u_t(-L) = u_t(L)$. One of the first integrals of the equation is then

$$\mathcal{I}[u] = \int_{-L}^{L} \frac{1}{2}u_t^2 + \frac{1}{2}u_x^2 + 1 - \cos(u)\mathrm{d}x.$$

Introducing $v = u_t$ to get a first order in time PDE,

$$\begin{bmatrix} u_t \\ v_t \end{bmatrix} = \begin{bmatrix} v \\ u_{xx} - \sin(u) \end{bmatrix},$$

we have the first integral

$$\mathcal{I}[u, v] = \int_{\mathbb{R}} \frac{1}{2} v^2 + \frac{1}{2} u_x^2 + 1 - \cos(u) \mathrm{d}x.$$

Finding the variational derivative of this, one can interpret the equation in the form (2.2.2) with S and $\frac{\delta \mathcal{I}}{\delta u}$ as follows:

$$S = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}, \quad \frac{\delta \mathcal{I}}{\delta u}[u, v] = \begin{bmatrix} \sin(u) - u_{xx} \\ v \end{bmatrix}.$$

Next, we approximate $\mathcal{I}$ by quadrature with points $\{x_i\}_{i=0}^N$ and weights $\{\kappa_i\}_{i=0}^N$, then applying central differences δ to approximate spatial derivatives. Periodic expansions are used at the endpoints.

$$\begin{aligned} \mathcal{I}[u, v] &\simeq \sum_{i=0}^{N} \kappa_i \left(\frac{1}{2} v_i^2 + \frac{1}{2} u_{x,i}^2 + 1 - \cos(u_i) \right) \\ &\simeq \sum_{i=0}^{N} \kappa_i \left(\frac{1}{2} v_i^2 + \frac{1}{2} \left(\frac{\delta u_i}{\delta x_i} \right)^2 + 1 - \cos(u_i) \right) := \mathcal{I}_{\mathbf{x}}(\mathbf{u}). \end{aligned}$$

The periodic boundary conditions are enforced by setting $u_0 = u_N$ and $v_0 = v_N$. The κ_i were chosen as the quadrature weights associated with the composite trapezoidal rule, i.e.

$$\kappa_0 = \frac{x_1 - x_0}{2}, \quad \kappa_N = \frac{x_N - x_{N-1}}{2}, \quad \kappa_i = \frac{x_{i+1} - x_{i-1}}{2}, i = 1, ..., N-1.$$

Furthermore, S was approximated by the matrix

$$S_d = \begin{bmatrix} 0 & I \\ -I & 0 \end{bmatrix},$$

where the I are $N \times N$ identity matrices. The exact solution considered was

$$u(x, t) = 4 \tan^{-1} \left(\frac{\sinh\left(\frac{ct}{\sqrt{1 - c^2}} \right)}{c \cosh\left(\frac{x}{\sqrt{1 - c^2}} \right)} \right).$$

This is a *kink-antikink* system, an interaction between two solitons, each moving in different directions with speed c, resulting in two wave fronts traveling in opposite directions. The wave fronts become steeper as $c \to 1$; in this test, we chose $c = 0.99$. The left hand plot of 2.1 shows the grid point placements over time when using DGMM. The areas of high grid point density are where the two fronts are located. The right hand plot shows the error after 200 time steps as a function of N. It is clear that using moving mesh methods increases the accuracy of the DG method drastically, to the point where it provides better solutions than the MP method. The MPMM method was unstable and did not produce any results except when using restrictively small time steps, so results using this are omitted here.

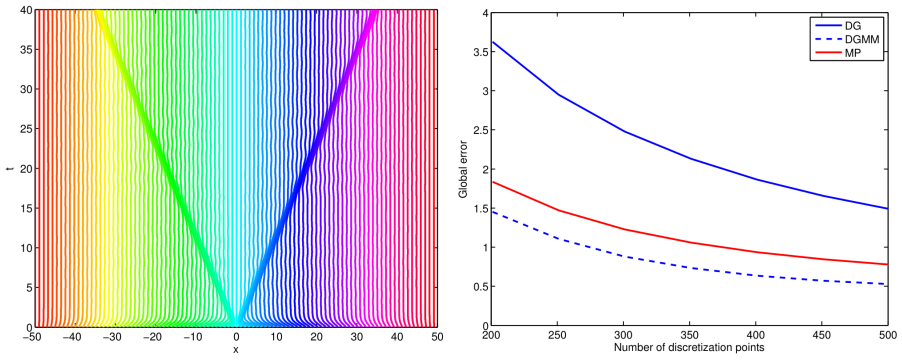


Figure 2.1: *Left:* Grid movement. Each line represents the path of one grid point in time. *Right:* Error at stopping time. $T = 8$, $\Delta t = 0.04$, $L = 15$. DG: Discrete Gradient. MP: MidPoint. MM: Moving Mesh.

Bibliography

[1] I. Babuška and B. Guo, *The h, p and h-p version of the finite element method; basis theory and applications*, Adv. Eng. Softw., 15 (1992), pp. 159–174.

[2] T. Bridges and S. Reich, *Multi-symplectic integrators: numerical schemes for Hamiltonian PDEs that conserve symplecticity*, Phys. Lett. A, 284 (2001), pp. 184–193.

[3] E. Celledoni, V. Grimm, R. I. McLachlan, D. I. McLaren, D. O'Neale, B. Owren, and G. R. W. Quispel, *Preserving energy resp. dissipation in numerical PDEs using the "average vector field" method*, J. Comput. Phys., 231 (2012), pp. 6770–6789.

[4] M. DAHLBY AND B. OWREN, *A general framework for deriving integral preserving numerical methods for PDEs*, SIAM J. Sci. Comput., 33 (2011), pp. 2318–2340.

[5] S. EIDNES, B. OWREN, AND T. RINGHOLM, *Adaptive energy preserving methods for partial differential equations*, Adv. Comput. Math., (2017).

[6] D. FURIHATA AND T. MATSUO, *Discrete variational derivative method*, Chapman & Hall/CRC Numerical Analysis and Scientific Computing, CRC Press, Boca Raton, FL, 2011.

[7] O. GONZALEZ, *Time integration and discrete Hamiltonian systems*, J. Nonlinear Sci., 6 (1996), pp. 449–467.

[8] E. HAIRER, C. LUBICH, AND G. WANNER, *Geometric numerical integration*, vol. 31 of Springer Series in Computational Mathematics, Springer, Heidelberg, 2010.

[9] W. HUANG AND R. RUSSELL, *Adaptive moving mesh methods*, vol. 174 of Springer Series in Applied Mathematical Sciences, Springer-Verlag, New York, 2010.

[10] R. I. MCLACHLAN, G. R. W. QUISPEL, AND N. ROBIDOUX, *Geometric integration using discrete gradients*, Philos. T. R. Soc. A, 357 (1999), pp. 1021–1045.

[11] Y. MIYATAKE AND T. MATSUO, *A note on the adaptive conservative/dissipative discretization for evolutionary partial differential equations*, J. Comput. Appl. Math., 274 (2015), pp. 79–87.

[12] T. YAGUCHI, T. MATSUO, AND M. SUGIHARA, *An extension of the discrete variational method to nonuniform grids*, J. Comput. Phys., 229 (2010), pp. 4382–4423.

[13] T. YAGUCHI, T. MATSUO, AND M. SUGIHARA, *The discrete variational derivative method based on discrete differential forms*, J. Comput. Phys., 231 (2012), pp. 3963–3986.

[14] P. A. ZEGELING, *r-refinement for evolutionary PDEs with finite elements or finite differences*, Appl. Numer. Math., 26 (1998), pp. 97–104.

Adaptive energy preserving methods for partial differential equations

Sølve Eidnes, Brynjulf Owren and Torbjørn Ringholm

Published in Advances in Computational Mathematics 44 (3), pp. 815-839 (2018)

Adaptive energy preserving methods for partial differential equations

Abstract. A framework for constructing integral preserving numerical schemes for time-dependent partial differential equations on non-uniform grids is presented. The approach can be used with both finite difference and partition of unity methods, thereby including finite element methods. The schemes are then extended to accommodate r-, h- and p-adaptivity. To illustrate the ideas, the method is applied to the Korteweg–de Vries equation and the sine-Gordon equation. Results from numerical experiments are presented.

3.1 Introduction

Difference schemes with conservation were introduced by Courant, Friedrichs and Lewy in [8], where a discrete conservation law for a finite difference approximation of the wave equation was derived. Their methods are often called energy methods [11] or energy-conserving methods [18], although the conserved quantity is often not energy in the physical sense. The primary motivation for developing conservative methods was originally to devise a norm that could guarantee global stability. This was still an objective, in addition to proving existence and uniqueness of solutions, when the energy methods garnered newfound interest in the 1950s and 1960s, resulting in new developments such as generalizations of the methods and more difference schemes, summarized by Richtmyer and Morton in [26].

In the 1970s, the motivation behind studying schemes that preserve invariant quantities changed, as the focus shifted to the conservation property itself. Li and Vu-Quoc presented in [18] a historical survey of conservative methods developed up to the early 1990s. They state that this line of work is motivated by the fact that in some situations, the success of a numerical solution will depend on its ability to preserve one or more of the invariant properties of the original differential equation. In addition, as noted in [7,14], there is the general idea that transferring more of the properties of the original continuous dynamical system over to a discrete dynamical system may lead to a more accurate numerical approximation of the solution, especially over long time intervals.

In recent years, there has been a greater interest in developing systematic techniques applicable to larger classes of differential equations. Hairer, Lubich and Wanner give in [14] a presentation of geometric integrators for differential equations, i.e. methods for solving ordinary differential equations (ODEs) that preserve a geometric structure of the system. Examples of such geometric structures are symplectic structures, symmetries, reversing symmetries, isospectrality, Lie group structure, orthonormality, first integrals, and other invariants,

such as volume and invariant measure.

In this paper we will be concerned with the preservation of first integrals of PDEs. From the ODE literature we find that the most general methods for preserving first integrals are tailored schemes, in the sense that the vector field of the ODE does not by itself provide sufficient information, so the schemes make explicit use of the first integral. An obvious approach in this respect is projection, where the solution is first advanced using any consistent numerical scheme and then this approximation is projected onto the appropriate level set of the invariant. In the same class of tailored methods one also has the discrete gradient methods, usually attributed to Gonzalez [13]. For the subclass of canonical Hamiltonian systems, the energy can be preserved by means of a general purpose method called the averaged vector field method, see e.g. [25].

The notion of discrete gradient methods for ordinary differential equations has a counterpart for partial differential equations called the discrete variational derivative method. Such schemes have been developed since the late 1990s in a number of articles by Japanese researchers such as Furihata, Matsuo, Sugihara, and Yaguchi. A relatively recent account of this work can be found in the monograph [12]. More recently, the development of integral preserving schemes for PDEs has been systematised and eased, in particular by using the aforementioned tools from ordinary differential equations, see for instance [6,9]. Most of the schemes one finds in the literature are based on a finite difference approach, and usually on fixed, uniform grids. There are however some exceptions. Yaguchi, Matsuo and Sugihara presented in [27,28] two different discrete variational derivative methods on fixed, non-uniform grids, specifically defined for certain classes of PDEs. Non-uniform grids are of particular importance for multidimensional problems, since the use of uniform grids will greatly restrict the types of domains possible to discretize. Another important consequence of being able to use non-uniform grids is that it allows for the use of time-adaptive spatial meshes for solving partial differential equations. Adaptive energy preserving schemes for the Korteweg–de Vries and Cahn–Hilliard equations have been developed recently [22] by Miyatake and Matsuo. The main objective of this paper is to propose a general framework for numerical methods for PDEs that combine mesh adaptivity with first integral conservation.

Several forms of adaptive methods exist, and they can roughly be categorized as r-, h- and p-adaptive. When applying r-adaptivity, one keeps the number of degrees of freedom constant while modifying the mesh at each time step to e.g. cluster in problematic areas such as boundary layers or to follow wave fronts. When applying the Finite Difference Method (FDM) or the Finite Element Method (FEM), moving mesh methods may be used for r-adaptivity, some examples of which may be found in [16, 17, 29]. When using Partition of Unity Methods (PUM) (and in particular when using FEM), h- and p-adaptivity

relate to adjusting the number of elements and the basis functions used on the elements, respectively. For PUM methods there exist strategies for h- and p-adaptivity based both on a priori and a posteriori error analysis [1]. Common to all of these strategies is that, based on estimated function values in preceding time steps, one can suggest improved discretization parameters for the next time step. In the FDM approach, these discretization parameters consist of the mesh points $\mathbf{x}$, while in the PUM approach the parameters encompass information about both the mesh and the basis functions. We will, in general, denote a collection of discretization parameters by $\mathbf{p}$, and assume that the discretization parameters are changed separately from the degrees of freedom $\mathbf{u}$ of the problem when using adaptive methods. That is, starting with an initial set of discretization parameters $\mathbf{p}^0$ and initial values $\mathbf{u}^0$, one first decides upon $\mathbf{p}^1$ before calculating $\mathbf{u}^1$, then finding $\mathbf{p}^2$, then $\mathbf{u}^2$, etc., in a decoupled fashion.

A first integral of a PDE is a functional $\mathcal{I}$ on an infinite-dimensional space, yet our numerical methods will reduce the problem to a finite-dimensional setting. Therefore, we cannot preserve the exact value of the first integral; instead, we will preserve a consistent approximation to the first integral, $\mathcal{I}_{\mathbf{p}}(\mathbf{u})$. The approximation will be dependent on the discretization parameters $\mathbf{p}$ and, since adaptivity alters the values of $\mathbf{p}$, we will therefore aim to preserve the value of the approximated first integral across all discretization parameters, i.e. we will require that $\mathcal{I}_{\mathbf{p}^{n+1}}(\mathbf{u}^{n+1}) = \mathcal{I}_{\mathbf{p}^n}(\mathbf{u}^n)$. Here, and in the following, superscripts denote time steps unless otherwise specified.

In this article, we present a method for developing adaptive numerical schemes that conserve an approximated first integral. In Section 2, the PDE problem is stated, and two classes of first integral preserving methods using arbitrary, constant discretization parameters are presented; one using an FDM approach and the other a PUM approach for spatial discretization. A connection to existing methods is then established. In Section 3, we present a way of adding adaptivity to the methods from Section 2 and the modifications needed to retain the first integral preservation property, before showing that certain projection methods form a subclass of the methods thus obtained. Section 4 contains examples of the methods applied to two PDEs and numerical results assessing the quality of the numerical solutions as compared to a standard implicit method.

3.2 Spatial discretization with fixed mesh

3.2.1 Problem statement

Consider a partial differential equation

$$u_t = f(\mathbf{x}, u^J), \qquad \mathbf{x} \in \Omega \subseteq \mathbb{R}^d, \quad u \in \mathcal{B} \subseteq L^2, \tag{3.2.1}$$

where u^J denotes u itself and its partial derivatives of any order with respect to the spatial variables $x_1, ..., x_d$. We shall not specify the space $\mathcal{B}$ further, but assume that it is sufficiently regular to allow all operations used in the following. For ease of reading, all t-dependence will be suppressed in the notation wherever it is irrelevant. Also, from here on, square brackets are used to denote dependence on a function and its partial derivatives of any order with respect to the independent variables t and $x_1, ..., x_d$. We recall the definition of the *variational derivative* of a functional $H[u]$ as the function $\frac{\delta H}{\delta u}[u]$ satisfying

$$\left\langle \frac{\delta H}{\delta u}[u], v \right\rangle_{L^2} = \frac{\mathrm{d}}{\mathrm{d}\epsilon}\bigg|_{\epsilon=0} H[u + \epsilon v] \quad \forall v \in \mathcal{B}, \tag{3.2.2}$$

and define a *first integral* of (3.2.1) to be a functional $\mathcal{I}[u]$ satisfying

$$\left\langle \frac{\delta \mathcal{I}}{\delta u}[u], f(\mathbf{x}, u^J) \right\rangle_{L^2} = 0, \quad \forall u \in \mathcal{B}.$$

We may observe that $\mathcal{I}[u]$ is preserved over time, since this implies

$$\frac{\mathrm{d}\mathcal{I}}{\mathrm{d}t} = \left\langle \frac{\delta \mathcal{I}}{\delta u}[u], \frac{\partial u}{\partial t} \right\rangle_{L^2} = 0.$$

Furthermore, we may observe that if there exists an operator $S(\mathbf{x}, u^J)$, skew-symmetric with respect to the L^2 inner product, such that

$$f(\mathbf{x}, u^J) = S(\mathbf{x}, u^J) \frac{\delta \mathcal{I}}{\delta u}[u],$$

then $\mathcal{I}[u]$ is a first integral of (3.2.1), and we can state (3.2.1) in the form

$$u_t = S(\mathbf{x}, u^J) \frac{\delta \mathcal{I}}{\delta u}[u]. \tag{3.2.3}$$

This can be considered as the PDE analogue of an ODE with a first integral, in which case we have a system

$$\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} = S(\mathbf{u}) \nabla_{\mathbf{u}} I(\mathbf{u}), \tag{3.2.4}$$

where $S(\mathbf{u})$ is a skew-symmetric matrix [20]. The gradient is defined as usual, but for clarity in later use we have added a subscript to specify that it is a vector of partial derivatives with respect to the coordinates of $\mathbf{u}$. Note that Hamiltonian equations are contained of this class of ODEs. For such differential equations, there exist numerical methods preserving the first integral $I(\mathbf{u})$, for instance the discrete gradient methods, which are of the form

$$\frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\Delta t} = \bar{S}(\mathbf{u}^n, \mathbf{u}^{n+1}) \overline{\nabla} I(\mathbf{u}^n, \mathbf{u}^{n+1}),$$

where $\bar{S}(\mathbf{u}^n, \mathbf{u}^{n+1})$ is a consistent skew-symmetric time-discrete approximation to $S(\mathbf{u})$ and $\overline{\nabla} I(\mathbf{v}, \mathbf{u})$ is a discrete gradient of $I(\mathbf{u})$, i.e. a function satisfying

$$(\overline{\nabla} I(\mathbf{v}, \mathbf{u}))^T (\mathbf{u} - \mathbf{v}) = I(\mathbf{u}) - I(\mathbf{v}), \tag{3.2.5}$$

$$\overline{\nabla} I(\mathbf{u}, \mathbf{u}) = \nabla_{\mathbf{u}} I(\mathbf{u}). \tag{3.2.6}$$

There are several possible choices of discrete gradients available, one of which is the Average Vector Field (AVF) discrete gradient [6], given by

$$\overline{\nabla} I(\mathbf{v}, \mathbf{u}) = \int_0^1 \nabla_{\mathbf{u}} I(\xi \mathbf{u} + (1 - \xi)\mathbf{v}) \mathrm{d}\xi,$$

which will be used for numerical experiments in the final chapter. Our approach to solving (3.2.1) on non-uniform grids is based upon considering the PDE in the form (3.2.3), reducing it to a system of ODEs of the form (3.2.4) and applying a discrete gradient method. This is done by finding a discrete approximation $\mathcal{I}_{\mathbf{p}}$ to $\mathcal{I}$ and using this to obtain a discretization in the spatial variables, which is achieved through either a finite difference approach or a variational approach.

3.2.2 Finite difference method

In the finite difference approach, we restrict ourselves to obtaining approximate values of u at the grid points $\mathbf{x}_0, ..., \mathbf{x}_M$, which can be interpreted as quadrature points with some associated nonzero quadrature weights $\kappa_0, ..., \kappa_M$. The grid points constitute the discretization parameters $\mathbf{p}$. We can then approximate the L^2 inner product by quadrature to arrive at a weighted inner product:

$$\langle u, v \rangle_{L^2} = \int_\Omega u(\mathbf{x}) v(\mathbf{x}) \mathrm{d}x \simeq \sum_{i=0}^{M} \kappa_i u(\mathbf{x}_i) v(\mathbf{x}_i) = \mathbf{u}^T D(\kappa) \mathbf{v} = \langle \mathbf{u}, \mathbf{v} \rangle_\kappa ,$$

where $D(\kappa) = \mathrm{diag}(\kappa_0, ..., \kappa_M)$. Assume that there exists a consistent approximation $\mathcal{I}_{\mathbf{p}}(\mathbf{u})$ to the functional $\mathcal{I}[u]$, dependent on the values of u at the points $\mathbf{x}_i$. Then, we can characterize the discretized variational derivative by asserting that

$$\left\langle \frac{\delta \mathcal{I}_{\mathbf{p}}}{\delta \mathbf{u}}(\mathbf{u}), \mathbf{v} \right\rangle_\kappa = \frac{\mathrm{d}}{\mathrm{d}\epsilon}\bigg|_{\epsilon=0} \mathcal{I}_{\mathbf{p}}(\mathbf{u} + \epsilon \mathbf{v}) \quad \forall \mathbf{v} \in \mathbb{R}^{M+1},$$

meaning

$$\left(\frac{\delta \mathcal{I}_{\mathbf{p}}}{\delta \mathbf{u}}(\mathbf{u}) \right)^T D(\kappa) \mathbf{v} = (\nabla_{\mathbf{u}} \mathcal{I}_{\mathbf{p}}(\mathbf{u}))^T \mathbf{v} \quad \forall \mathbf{v} \in \mathbb{R}^{M+1},$$

from which we conclude that

$$\frac{\delta \mathcal{I}_{\mathbf{p}}}{\delta \mathbf{u}}(\mathbf{u}) = D(\kappa)^{-1} \nabla_{\mathbf{u}} \mathcal{I}_{\mathbf{p}}(\mathbf{u}). \tag{3.2.7}$$

Using this as a discretization of $\frac{\delta \mathcal{I}}{\delta u}[u]$ and approximating $S(\mathbf{x}, u^J)$ by a matrix $S_d(\mathbf{u})$, skew-symmetric with respect to $\langle \cdot, \cdot \rangle_\kappa$, we obtain a discretization of (3.2.3) as:

$$\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} = S_{\mathbf{p}}(\mathbf{u}) \nabla_{\mathbf{u}} \mathcal{I}_{\mathbf{p}}(\mathbf{u}), \tag{3.2.8}$$

where $S_{\mathbf{p}}(\mathbf{u}) = S_d(\mathbf{u}) D(\kappa)^{-1}$. This system of ODEs is of the form (3.2.4), since

$$\begin{aligned} S_{\mathbf{p}}(\mathbf{u})^T &= (S_d(\mathbf{u}) D(\kappa)^{-1})^T \\ &= D(\kappa)^{-1} S_d(\mathbf{u})^T D(\kappa) D(\kappa)^{-1} \\ &= -D(\kappa)^{-1} D(\kappa) S_d(\mathbf{u}) D(\kappa)^{-1} \\ &= -S_d(\mathbf{u}) D(\kappa)^{-1} \\ &= -S_{\mathbf{p}}(\mathbf{u}). \end{aligned}$$

This allows us to apply first integral preserving methods for systems of ODEs to solve the spatially discretized system. For example, we may consider using a discrete gradient $\overline{\nabla} \mathcal{I}_{\mathbf{p}}$, and a skew-symmetric, time-discrete approximation $S_{\mathbf{p}}(\mathbf{u}^n, \mathbf{u}^{n+1})$ to $S_{\mathbf{p}}(\mathbf{u})$, where $\mathbf{u}^n = \mathbf{u}(t_n)$, $t_n = n\Delta t$. Then, the following scheme will preserve the approximated first integral $\mathcal{I}_{\mathbf{p}}$ in the sense that $\mathcal{I}_{\mathbf{p}}(\mathbf{u}^{n+1}) = \mathcal{I}_{\mathbf{p}}(\mathbf{u}^n)$:

$$\frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\Delta t} = S_{\mathbf{p}}(\mathbf{u}^n, \mathbf{u}^{n+1}) \overline{\nabla} \mathcal{I}_{\mathbf{p}}(\mathbf{u}^n, \mathbf{u}^{n+1}). \tag{3.2.9}$$

3.2.3 Partition of unity method

One may also approach the problem of spatially discretizing the PDE through the use of variational methods such as the Partition of Unity Method (PUM) [21], which generalizes the Finite Element Method (FEM). Here, the variational structure of the functional derivative can be utilized in a natural way, such that one avoids having to approximate $S(\mathbf{x}, u^J)$. We begin by stating a weak form of (3.2.3). Then, the problem consists of finding $u \in \mathcal{B}$ such that

$$\left\langle u_t, v \right\rangle_{L^2} = \left\langle S(\mathbf{x}, u^J) \frac{\delta \mathcal{I}}{\delta u}[u], v \right\rangle_{L^2} = -\left\langle \frac{\delta \mathcal{I}}{\delta u}[u], S(\mathbf{x}, u^J) v \right\rangle_{L^2} \qquad \forall v \in \mathcal{B}. \tag{3.2.10}$$

Employing a Galerkin formulation, we restrict the search to a finite dimensional subspace $\mathcal{B}^h = \text{span}\{\varphi_0, ...\varphi_M\} \subseteq \mathcal{B}$, and approximate u by the function

$$u^h(x,t) = \sum_{i=0}^{M} u_i(t)\varphi_i(x).$$

We denote by $\mathbf{p}$ the collection of discretization parameters defining $\mathcal{B}^h$; this includes information about mesh points, element types and shapes of basis functions. Furthermore, we define the canonical mapping $\Phi_{\mathbf{p}} : \mathbb{R}^{M+1} \to \mathcal{B}^h$ given by

$$\Phi_{\mathbf{p}}(\mathbf{u}) = \sum_{i=0}^{M} u_i\varphi_i, \tag{3.2.11}$$

and the discrete first integral $\mathcal{I}_{\mathbf{p}}$ by

$$\mathcal{I}_{\mathbf{p}}(\mathbf{u}) = \mathcal{I}(\Phi_{\mathbf{p}}(\mathbf{u})).$$

The following lemma will prove useful later in the construction of the method:

Lemma 3.1. *For any $u^h, v \in \mathcal{B}^h$,*

$$\left.\frac{\mathrm{d}}{\mathrm{d}\epsilon}\right|_{\epsilon=0} \mathcal{I}(u^h + \epsilon v) = (\nabla_{\mathbf{u}}\mathcal{I}_{\mathbf{p}}(\mathbf{u}))^T\mathbf{v}.$$

Proof.

$$\begin{aligned}
\left.\frac{\mathrm{d}}{\mathrm{d}\epsilon}\right|_{\epsilon=0} \mathcal{I}(u^h + \epsilon v) &= \left.\frac{\mathrm{d}}{\mathrm{d}\epsilon}\right|_{\epsilon=0} \mathcal{I}(\Phi_{\mathbf{p}}(\mathbf{u} + \epsilon\mathbf{v})) \\
&= \left.\left\langle \frac{\delta\mathcal{I}}{\delta u}[\Phi_{\mathbf{p}}(\mathbf{u}+\epsilon\mathbf{v})], \frac{\mathrm{d}}{\mathrm{d}\epsilon}\Phi_{\mathbf{p}}(\mathbf{u}+\epsilon\mathbf{v}) \right\rangle_{L^2}\right|_{\epsilon=0} \\
&= \left.\left\langle \frac{\delta\mathcal{I}}{\delta u}[\Phi_{\mathbf{p}}(\mathbf{u}+\epsilon\mathbf{v})], (\nabla_{\mathbf{u}}\Phi_{\mathbf{p}}(\mathbf{u}+\epsilon\mathbf{v}))^T\mathbf{v} \right\rangle_{L^2}\right|_{\epsilon=0} \\
&= \left\langle \frac{\delta\mathcal{I}}{\delta u}[\Phi_{\mathbf{p}}(\mathbf{u})], (\nabla_{\mathbf{u}}\Phi_{\mathbf{p}}(\mathbf{u}))^T\mathbf{v} \right\rangle_{L^2} \\
&= \sum_{i=0}^{M} v_i \left\langle \frac{\delta\mathcal{I}}{\delta u}[\Phi_{\mathbf{p}}(\mathbf{u})], \frac{\partial}{\partial u_i}\Phi_{\mathbf{p}}(\mathbf{u}) \right\rangle_{L^2} \\
&= \sum_{i=0}^{M} v_i \frac{\partial}{\partial u_i}\mathcal{I}[\Phi_{\mathbf{p}}(\mathbf{u})] = \sum_{i=0}^{M} v_i \frac{\partial}{\partial u_i}\mathcal{I}_{\mathbf{p}}(\mathbf{u}) = (\nabla_{\mathbf{u}}\mathcal{I}_{\mathbf{p}}(\mathbf{u}))^T\mathbf{v}.
\end{aligned}$$

□

We observe that for $u, v \in \mathcal{B}^h$, the L^2 inner product has a discrete counterpart:

$$\langle u, v \rangle_{L^2} = \sum_{i=0}^{M} \sum_{j=0}^{M} u_i v_j \left\langle \varphi_i, \varphi_j \right\rangle_{L^2} = \mathbf{u}^T A \mathbf{v} = \langle \mathbf{u}, \mathbf{v} \rangle_A$$

with the symmetric positive definite matrix A given by $A_{ij} = \left\langle \varphi_i, \varphi_j \right\rangle_{L^2}$. Note also that equation (3.2.10) is satisfied in $\mathcal{B}^h$ if it is satisfied for all basis functions φ_j. The Galerkin form of the problem therefore consists of finding $u_i(t)$ such that

$$\sum_{i=0}^{M} \frac{\mathrm{d}u_i}{\mathrm{d}t} \left\langle \varphi_i, \varphi_j \right\rangle_{L^2} = -\left\langle \frac{\delta \mathcal{I}}{\delta u}[u^h], S(\mathbf{x}, u^{h,J}) \varphi_j \right\rangle_{L^2} \qquad \forall j \in \{0, ..., M\}. \quad (3.2.12)$$

This weak form is rather unwieldy and does not give rise to a system of the form (3.2.4), so in order to make further progress, we consider the projection of $\frac{\delta \mathcal{I}}{\delta u}[u^h]$ onto $\mathcal{B}^h$:

$$\frac{\delta \mathcal{I}}{\delta u}^h [u^h] = \sum_{i=0}^{M} w_i^h[u^h] \varphi_i(x) = \sum_{i=0}^{M} w_i(\mathbf{u}) \varphi_i(x),$$

where $w_i(\mathbf{u}) = w_i^h[\Phi(\mathbf{u})] = w_i^h[u^h]$ are coefficients that will be characterized later. Replacing $\frac{\delta \mathcal{I}}{\delta u}[u^h]$ by its projection in (3.2.12) gives the approximate weak form:

$$\sum_{i=0}^{M} \frac{\mathrm{d}u_i}{\mathrm{d}t} \left\langle \varphi_i, \varphi_j \right\rangle_{L^2} = -\sum_{i=0}^{M} w_i(\mathbf{u}) \left\langle \varphi_i, S(\mathbf{x}, u^{h,J}) \varphi_j \right\rangle_{L^2} \qquad \forall j \in \{0, ..., M\}.$$

Thus, we obtain a system of equations for the coefficients u_i:

$$A \frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} = -B(\mathbf{u}) \mathbf{w}(\mathbf{u}), \quad (3.2.13)$$

with the skew-symmetric matrix $B(\mathbf{u})$ given by $B(\mathbf{u})_{ji} = \left\langle \varphi_i, S(\mathbf{x}, \Phi(\mathbf{u})^J) \varphi_j \right\rangle_{L^2}$. Furthermore, we may characterize the vector $\mathbf{w}(\mathbf{u})$ by the following argument:

$$\begin{aligned} \mathbf{w}(\mathbf{u})^T A \mathbf{v} &= \left\langle \frac{\delta \mathcal{I}}{\delta u}^h [u^h], v \right\rangle_{L^2} = \left\langle \frac{\delta \mathcal{I}}{\delta u}[u^h], v \right\rangle_{L^2} \\ &= \frac{\mathrm{d}}{\mathrm{d}\epsilon}\bigg|_{\epsilon=0} \mathcal{I}(u^h + \epsilon v) = (\nabla_{\mathbf{u}} \mathcal{I}_{\mathbf{p}}(\mathbf{u}))^T \mathbf{v}, \end{aligned}$$

where the last equality holds by Lemma 3.1. This holds for all $\mathbf{v} \in \mathbb{R}^{M+1}$, and thus

$$\mathbf{w}(\mathbf{u}) = A^{-1} \nabla_{\mathbf{u}} \mathcal{I}_{\mathbf{p}}(\mathbf{u}). \quad (3.2.14)$$

Inserting (3.2.14) into (3.2.13) and left-multiplying by A^{-1}, we are left with an ODE for the coefficients u_i:

$$\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} = S_{\mathbf{p}}(\mathbf{u})\nabla_{\mathbf{u}}\mathcal{I}_{\mathbf{p}}(\mathbf{u}). \tag{3.2.15}$$

Here, $S_{\mathbf{p}}(\mathbf{u}) = -A^{-1}B(\mathbf{u})A^{-1}$ is a skew-symmetric matrix, and the system is thereby of the form (3.2.4), meaning $\mathcal{I}_{\mathbf{p}}$ can be preserved numerically using e.g. discrete gradient methods as in equation (3.2.9).

3.2.4 Discrete variational derivative methods

Let us now define a general framework for the discrete variational derivative methods that encompass the methods presented by Furihata, Matsuo and coauthors in a number of publications including [10–12, 27, 28].

Definition 3.1. Let $\mathcal{I}_{\mathbf{p}}$ be a consistent approximation to the functional $\mathcal{I}[u]$ discretized on $\mathbf{p}$ given by grid points $\mathbf{x}_i$ and quadrature weights κ_i, $i = 0, ..., M$. Then $\frac{\delta\mathcal{I}_{\mathbf{p}}}{\delta(\mathbf{v},\mathbf{u})}(\mathbf{v},\mathbf{u})$ is a discrete variational derivative of $\mathcal{I}_{\mathbf{p}}(\mathbf{u})$ if it is a continuous function satisfying

$$\left\langle \frac{\delta\mathcal{I}_{\mathbf{p}}}{\delta(\mathbf{v},\mathbf{u})}, \mathbf{u}-\mathbf{v} \right\rangle_{\kappa} = \mathcal{I}_{\mathbf{p}}(\mathbf{u}) - \mathcal{I}_{\mathbf{p}}(\mathbf{v}), \tag{3.2.16}$$

$$\frac{\delta\mathcal{I}_{\mathbf{p}}}{\delta(\mathbf{u},\mathbf{u})} = \frac{\delta\mathcal{I}_{\mathbf{p}}}{\delta\mathbf{u}}(\mathbf{u}), \tag{3.2.17}$$

and the discrete variational derivative methods for solving PDEs on the form (3.2.3) are given by

$$\frac{\mathbf{u}^{n+1}-\mathbf{u}^n}{\Delta t} = S_d(\mathbf{u}^n, \mathbf{u}^{n+1})\frac{\delta\mathcal{I}_{\mathbf{p}}}{\delta(\mathbf{u}^n, \mathbf{u}^{n+1})}, \tag{3.2.18}$$

where $S_d(\mathbf{u}^n, \mathbf{u}^{n+1})$ is a time-discrete approximation to $S_d(\mathbf{u})$, skew-symmetric with respect to the inner product $\langle\cdot,\cdot\rangle_{\kappa}$.

Proposition 3.1. *A discrete gradient method (3.2.9) applied to the system of ODEs (3.2.8) or (3.2.15) is equivalent to a discrete variational derivative method as given by (3.2.18), with*

$$S_d(\mathbf{u}^n, \mathbf{u}^{n+1}) = S_{\mathbf{p}}(\mathbf{u}^n, \mathbf{u}^{n+1})D(\kappa),$$

and the discrete variational derivative

$$\frac{\delta\mathcal{I}_{\mathbf{p}}}{\delta(\mathbf{v},\mathbf{u})} = D(\kappa)^{-1}\overline{\nabla}\mathcal{I}_{\mathbf{p}}(\mathbf{v},\mathbf{u}) \tag{3.2.19}$$

satisfying (3.2.16)-(3.2.17).

Proof. Applying (3.2.5), we find, for the discrete variational derivative (3.2.19),

$$\begin{aligned}\left\langle \frac{\delta \mathcal{I}_{\mathbf{p}}}{\delta(\mathbf{v},\mathbf{u})}, \mathbf{u}-\mathbf{v} \right\rangle_{\kappa} &= \left\langle D(\kappa)^{-1}\overline{\nabla}\mathcal{I}_{\mathbf{p}}(\mathbf{v},\mathbf{u}), \mathbf{u}-\mathbf{v} \right\rangle_{\kappa} \\ &= \left(D(\kappa)^{-1}\overline{\nabla}\mathcal{I}_{\mathbf{p}}\left(\mathbf{v},\mathbf{u}\right)\right)^{\mathrm{T}} D(\kappa)\left(\mathbf{u}-\mathbf{v}\right) \\ &= \overline{\nabla}\mathcal{I}_{\mathbf{p}}\left(\mathbf{v},\mathbf{u}\right)^{\mathrm{T}}\left(\mathbf{u}-\mathbf{v}\right) = \mathcal{I}_{\mathbf{p}}(\mathbf{u}) - \mathcal{I}_{\mathbf{p}}(\mathbf{v}),\end{aligned}$$

and hence (3.2.16) is satisfied. Furthermore, applying (3.2.6) and (3.2.7),

$$\frac{\delta \mathcal{I}_{\mathbf{p}}}{\delta(\mathbf{u},\mathbf{u})} = D(\kappa)^{-1}\overline{\nabla}\mathcal{I}_{\mathbf{p}}(\mathbf{u},\mathbf{u}) = D(\kappa)^{-1}\nabla_{\mathbf{u}}\mathcal{I}_{\mathbf{p}}(\mathbf{u}) = \frac{\delta \mathcal{I}_{\mathbf{p}}}{\delta \mathbf{u}}(\mathbf{u})$$

and (3.2.17) is also satisfied. □

Hence, all discrete variational derivative methods as given by (3.2.18) can be expressed as discrete gradient methods on the system of ODEs (3.2.8) or (3.2.15) obtained by discretizing (3.2.3) in space, and vice versa.

3.3 Adaptive discretization

3.3.1 Mapping solutions between parameter sets

Assuming that adaptive strategies are employed, one would obtain a new set of discretization parameters $\mathbf{p}$ at each time step. After such a $\mathbf{p}$ has been found, the solution using the previous parameters must be transferred to the new parameter set before advancing to the next time step. This transfer procedure can be done in either a preserving or a non-preserving manner. Let $\mathbf{p}^n$, $\mathbf{u}^n$, $\mathbf{p}^{n+1}$ and $\mathbf{u}^{n+1}$ denote the discretization parameters and the numerical values obtained at the current time step and next time step, respectively. Also, let $\hat{\mathbf{u}}$ denote the values of $\mathbf{u}^n$ transferred onto $\mathbf{p}^{n+1}$ by whatever means. We call the transfer operation preserving if $\mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}) = \mathcal{I}_{\mathbf{p}^n}(\mathbf{u}^n)$. If the transfer is preserving, then the next time step can be taken with a preserving scheme, e.g.

$$\frac{\mathbf{u}^{n+1}-\hat{\mathbf{u}}}{\Delta t} = S_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}},\mathbf{u}^{n+1})\overline{\nabla}\mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}},\mathbf{u}^{n+1}),$$

which is preserving in the sense that

$$\begin{aligned}\mathcal{I}_{\mathbf{p}^{n+1}}(\mathbf{u}^{n+1}) - \mathcal{I}_{\mathbf{p}^n}(\mathbf{u}^n) &= \mathcal{I}_{\mathbf{p}^{n+1}}(\mathbf{u}^{n+1}) - \mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}) \\ &= \left\langle \overline{\nabla}\mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}},\mathbf{u}^{n+1}), \mathbf{u}^{n+1}-\hat{\mathbf{u}} \right\rangle \\ &= \Delta t \left\langle \overline{\nabla}\mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}},\mathbf{u}^{n+1}), S_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}},\mathbf{u}^{n+1})\overline{\nabla}\mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}},\mathbf{u}^{n+1}) \right\rangle \\ &= 0,\end{aligned}$$

since $S_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}, \mathbf{u}^{n+1})$ is skew-symmetric. If non-preserving transfer is used, corrections are needed in order to obtain a preserving numerical method.

Proposition 3.2. *The scheme*

$$\mathbf{u}^{n+1} = \hat{\mathbf{u}} - \frac{(\mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}) - \mathcal{I}_{\mathbf{p}^n}(\mathbf{u}^n))\mathbf{z}}{\left\langle \overline{\nabla}\mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}, \mathbf{u}^{n+1}), \mathbf{z} \right\rangle} + \Delta t S_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}, \mathbf{u}^{n+1}) \overline{\nabla}\mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}, \mathbf{u}^{n+1}), \quad (3.3.1)$$

where $\mathbf{z}$ *is an arbitrary vector chosen such that* $\left\langle \overline{\nabla}\mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}, \mathbf{u}^{n+1}), \mathbf{z} \right\rangle \neq 0$*, is first integral preserving in the sense that* $\mathcal{I}_{\mathbf{p}^{n+1}}(\mathbf{u}^{n+1}) - \mathcal{I}_{\mathbf{p}^n}(\mathbf{u}^n) = 0$.

Proof.

$$\begin{aligned}
\mathcal{I}_{\mathbf{p}^{n+1}}(\mathbf{u}^{n+1}) - \mathcal{I}_{\mathbf{p}^n}(\mathbf{u}^n) &= \mathcal{I}_{\mathbf{p}^{n+1}}(\mathbf{u}^{n+1}) - \mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}) + \mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}) - \mathcal{I}_{\mathbf{p}^n}(\mathbf{u}^n) \\
&= \left\langle \overline{\nabla}\mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}, \mathbf{u}^{n+1}), \mathbf{u}^{n+1} - \hat{\mathbf{u}} \right\rangle + \mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}) - \mathcal{I}_{\mathbf{p}^n}(\mathbf{u}^n) \\
&= \left\langle \overline{\nabla}\mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}, \mathbf{u}^{n+1}), \mathbf{u}^{n+1} - \hat{\mathbf{u}} + \frac{(\mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}) - \mathcal{I}_{\mathbf{p}^n}(\mathbf{u}^n))\mathbf{z}}{\left\langle \overline{\nabla}\mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}, \mathbf{u}^{n+1}), \mathbf{z} \right\rangle} \right\rangle \\
&= \Delta t \left\langle \overline{\nabla}\mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}, \mathbf{u}^{n+1}), S_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}, \mathbf{u}^{n+1}) \overline{\nabla}\mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}, \mathbf{u}^{n+1}) \right\rangle \\
&= 0.
\end{aligned}$$

The second equality follows from (3.2.5), the fourth equality from the scheme (3.3.1), and the last equality follows from the skew-symmetry of $S_{\mathbf{p}^{n+1}}$. □

The correcting direction $\mathbf{z}$ should be chosen so as to obtain a minimal correction, and such that $\langle \overline{\nabla}\mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}, \mathbf{u}^{n+1}), \mathbf{z} \rangle \neq 0$. One possibility is simply taking $\mathbf{z} = \overline{\nabla}\mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}, \mathbf{u}^{n+1})$. In the FDM case one may alternatively choose $\mathbf{z} = D(\kappa)^{-1}\overline{\nabla}\mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}, \mathbf{u}^{n+1})$, and in the PUM case, $\mathbf{z} = A^{-1}\overline{\nabla}\mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}, \mathbf{u}^{n+1})$.

When using the PUM formulation, one may obtain a method for preserving transfer in the following manner. Any changes through e.g. r- p- and/or h-refinement between time steps will result in a change in the shape and/or number of basis functions. Denote by $\mathcal{B}^h = \text{span}\{\varphi_i\}_{i=0}^{M}$ the trial space from the current time step and by $\hat{\mathcal{B}}^h = \text{span}\{\hat{\varphi}_i\}_{i=0}^{\hat{M}}$ the trial space for the next time step, and note that in general, $M \neq \hat{M}$. We do not concern ourselves with how the new basis is found, but simply acknowledge that the basis changes through adaptivity measures as presented in e.g. [16] or [1]. Our task is now to transfer the approximation u^h from $\mathcal{B}^h$ to $\hat{\mathcal{B}}^h$, obtaining an approximation $\hat{u}^h$, while conserving the first integral, i.e. $\mathcal{I}[u^h] = \mathcal{I}[\hat{u}^h]$. This can be formulated as a constrained minimization problem:

$$\min_{\hat{u}^h \in \hat{\mathcal{B}}^h} ||\hat{u}^h - u^h||_{L^2}^2 \quad \text{s.t.} \quad \mathcal{I}[\hat{u}^h] = \mathcal{I}[u^h].$$

We observe that

$$\begin{aligned}||\hat{u}^h - u^h||_{L^2}^2 &= \sum_{i=0}^{\hat{M}}\sum_{j=0}^{\hat{M}} \hat{u}_i \hat{u}_j \hat{A}_{ij} - 2\sum_{i=0}^{\hat{M}}\sum_{j=0}^{M} \hat{u}_i u_j^n C_{ij} + \sum_{i=0}^{M}\sum_{i=0}^{M} u_i^n u_j^n A_{ij} \\ &= \hat{\mathbf{u}}^T \hat{A}\hat{\mathbf{u}} - 2\hat{\mathbf{u}}^T C\mathbf{u}^n + \mathbf{u}^n A\mathbf{u}^n,\end{aligned}$$

where $A_{ij} = \langle \varphi_i, \varphi_j \rangle_{L^2}$, $\hat{A}_{ij} = \langle \hat{\varphi}_i, \hat{\varphi}_j \rangle_{L^2}$ and $C_{ij} = \langle \hat{\varphi}_i, \varphi_j \rangle_{L^2}$. Also observing that

$$\mathcal{I}[\hat{u}^h] = \mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}), \quad \mathcal{I}[u^h] = \mathcal{I}_{\mathbf{p}^n}(\mathbf{u}^n),$$

the problem can be reformulated as

$$\min_{\hat{\mathbf{u}} \in \mathbb{R}^{\hat{M}+1}} \hat{\mathbf{u}}^T \hat{A}\hat{\mathbf{u}} - 2\hat{\mathbf{u}}^T C\mathbf{u}^n + \mathbf{u}^n A\mathbf{u}^n \quad \text{s.t.} \quad \mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}) - \mathcal{I}_{\mathbf{p}^n}(\mathbf{u}^n) = 0.$$

This is a quadratic minimization problem with one nonlinear equality constraint. Using the method of Lagrange multipliers, we find $\hat{\mathbf{u}}$ as the solution of the nonlinear system of equations

$$\begin{aligned}\hat{A}\hat{\mathbf{u}} - C\mathbf{u}^n - \lambda \nabla_{\hat{\mathbf{u}}} \mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}) &= 0 \\ \mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}) - \mathcal{I}_{\mathbf{p}^n}(\mathbf{u}^n) &= 0,\end{aligned}$$

which can be solved numerically using a suitable nonlinear solver.

In general, applicable also in the FDM case, given $\bar{\mathbf{u}}$ obtained by interpolating $\mathbf{u}^n$ onto $\mathbf{p}^{n+1}$ in a non-preserving manner, a preserving transfer operation is obtained by solving the system of equations

$$\begin{aligned}\hat{\mathbf{u}} - \bar{\mathbf{u}} - \lambda \nabla_{\hat{\mathbf{u}}} \mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}) &= 0 \\ \mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}) - \mathcal{I}_{\mathbf{p}^n}(\mathbf{u}^n) &= 0.\end{aligned}$$

3.3.2 Projection methods

Let the function $f_{\mathbf{p}} : \mathbb{R}^M \times \mathbb{R}^M \to \mathbb{R}^M$ be such that

$$\frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\Delta t} = f_{\mathbf{p}}(\mathbf{u}^n, \mathbf{u}^{n+1}) \tag{3.3.2}$$

defines a step from time t_n to time t_{n+1} of any one-step method applied to (3.2.1) on the fixed grid represented by the discretization parameters $\mathbf{p}$. Then we define one step of an integral preserving linear projection method $\mathbf{u}^n \mapsto \mathbf{u}^{n+1}$ from $\mathbf{p}^n$ to $\mathbf{p}^{n+1}$ by

1. Interpolate $\mathbf{u}^n$ onto $\mathbf{p}^{n+1}$ by whatever means to get $\hat{\mathbf{u}}$,
2. Integrate $\hat{\mathbf{u}}$ one time step by computing $\tilde{\mathbf{u}} = \hat{\mathbf{u}} + \Delta t f_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}, \tilde{\mathbf{u}})$,

3. Compute $\mathbf{u}^{n+1}$ by solving the system of $M+1$ equations $\mathbf{u}^{n+1} = \tilde{\mathbf{u}} + \lambda \mathbf{z}$ and $\mathcal{I}_{\mathbf{p}^{n+1}}(\mathbf{u}^{n+1}) = \mathcal{I}_{\mathbf{p}^n}(\mathbf{u}^n)$, for $\mathbf{u}^{n+1} \in \mathbb{R}^M$ and $\lambda \in \mathbb{R}$, where the direction of projection $\mathbf{z}$ is typically an approximation to $\nabla_{\mathbf{u}} \mathcal{I}_{\mathbf{p}^{n+1}}(\mathbf{u}^{n+1})$.

By utilizing the fact that for a method defined by (3.3.2) there exists an implicitly defined map $\Psi_{\mathbf{p}} : \mathbb{R}^M \to \mathbb{R}^M$ such that $\mathbf{u}^{n+1} = \Psi_{\mathbf{p}} \mathbf{u}^n$, we define

$$g_{\mathbf{p}}(\mathbf{u}^n) := \frac{\Psi_{\mathbf{p}} \mathbf{u}^n - \mathbf{u}^n}{\Delta t},$$

and may then write the tree points above in an equivalent, more compact form as: Compute $\mathbf{u}^{n+1} \in \mathbb{R}^M$ and $\lambda \in \mathbb{R}$ such that

$$\mathbf{u}^{n+1} - \hat{\mathbf{u}} - \Delta t\, g_{\mathbf{p}^{n+1}}\left(\hat{\mathbf{u}}\right) - \lambda \mathbf{z} = 0, \tag{3.3.3}$$

$$\mathcal{I}_{\mathbf{p}^{n+1}}(\mathbf{u}^{n+1}) - \mathcal{I}_{\mathbf{p}^n}(\mathbf{u}^n) = 0, \tag{3.3.4}$$

where $\hat{\mathbf{u}}$ is $\mathbf{u}^n$ interpolated onto $\mathbf{p}^{n+1}$ by an arbitrary procedure.

The following theorem and proof are reminiscent of Theorem 2 and its proof in [23], whose subsequent corollary shows how linear projection methods for solving ODEs are a subset of discrete gradient methods.

Theorem 3.1. *Let $g_{\mathbf{p}} : \mathbb{R}^M \to \mathbb{R}^M$ be a consistent discrete approximation of f in (3.2.1) and let $\overline{\nabla}\mathcal{I}_{\mathbf{p}}(\mathbf{u}^n, \mathbf{u}^{n+1})$ be any discrete gradient of the consistent approximation $\mathcal{I}_{\mathbf{p}}(\mathbf{u})$ of $\mathcal{I}[u]$ defined by (3.2.2) on the grid given by discretization parameters $\mathbf{p}$. If we set $S_{\mathbf{p}^{n+1}}$ in (3.3.1) to be*

$$S_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}, \mathbf{u}^{n+1}) = \frac{g_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}})\mathbf{z}^T - \mathbf{z} g_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}})^T}{\left\langle \overline{\nabla}\mathcal{I}_{\mathbf{p}^{n+1}}\left(\hat{\mathbf{u}}, \mathbf{u}^{n+1}\right), \mathbf{z} \right\rangle}, \tag{3.3.5}$$

then the linear projection method for solving PDEs on a moving grid, given by (3.3.3)-(3.3.4), is equivalent to the discrete gradient method on moving grids, as given by (3.3.1).

Proof. For better readability, take $\overline{\nabla}\mathcal{I} := \overline{\nabla}\mathcal{I}_{\mathbf{p}^{n+1}}\left(\hat{\mathbf{u}}, \mathbf{u}^{n+1}\right)$. Assume that (3.3.3)-(3.3.4) are satisfied. By applying (3.3.4), we get that

$$\begin{aligned}
\mathcal{I}_{\mathbf{p}^n}(\mathbf{u}^n) - \mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}) &= \mathcal{I}_{\mathbf{p}^{n+1}}(\mathbf{u}^{n+1}) - \mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}) \\
&= \left\langle \overline{\nabla}\mathcal{I}, \mathbf{u}^{n+1} - \hat{\mathbf{u}} \right\rangle \\
&= \Delta t \left\langle \overline{\nabla}\mathcal{I}, g_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}) \right\rangle + \lambda \left\langle \overline{\nabla}\mathcal{I}, \mathbf{z} \right\rangle,
\end{aligned}$$

and hence

$$\lambda = \frac{\mathcal{I}_{\mathbf{p}^n}(\mathbf{u}^n) - \mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}})}{\left\langle \overline{\nabla}\mathcal{I}, \mathbf{z} \right\rangle} - \Delta t \frac{\left\langle \overline{\nabla}\mathcal{I}, g_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}) \right\rangle}{\left\langle \overline{\nabla}\mathcal{I}, \mathbf{z} \right\rangle} \tag{3.3.6}$$

Substituting this into (3.3.3), we get

$$\mathbf{u}^{n+1} = \hat{\mathbf{u}} + \frac{\mathcal{I}_{\mathbf{p}^n}(\mathbf{u}^n) - \mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}})}{\left\langle \overline{\nabla}\mathcal{I}, \mathbf{z} \right\rangle}\mathbf{z} + \Delta t \left(g_{\mathbf{p}^{n+1}}\left(\hat{\mathbf{u}}\right) - \frac{\left\langle \overline{\nabla}\mathcal{I}, g_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}) \right\rangle}{\left\langle \overline{\nabla}\mathcal{I}, \mathbf{z} \right\rangle}\mathbf{z} \right),$$

where

$$\begin{aligned} g_{\mathbf{p}^{n+1}}\left(\hat{\mathbf{u}}\right) - \frac{\left\langle \overline{\nabla}\mathcal{I}, g_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}) \right\rangle}{\left\langle \overline{\nabla}\mathcal{I}, \mathbf{z} \right\rangle}\mathbf{z} &= \frac{\overline{\nabla}\mathcal{I}^{\mathrm{T}}\mathbf{z} g_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}) - \overline{\nabla}\mathcal{I}^{\mathrm{T}} g_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}})\mathbf{z}}{\left\langle \overline{\nabla}\mathcal{I}, \mathbf{z} \right\rangle} \\ &= \frac{g_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}})\mathbf{z}^{\mathrm{T}}\overline{\nabla}\mathcal{I} - \mathbf{z} g_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}})^{\mathrm{T}}\overline{\nabla}\mathcal{I}}{\left\langle \overline{\nabla}\mathcal{I}, \mathbf{z} \right\rangle} \end{aligned}$$

and thus (3.3.1) is satisfied, with $S_{\mathbf{p}^{n+1}}$ as given by (3.3.5). Conversely, if $\mathbf{u}^{n+1}$ satisfies (3.3.1), then (3.3.4) is satisfied. Furthermore, inserting (3.3.5) into (3.3.1) and following the above deduction backwards, we get (3.3.3), with λ defined by (3.3.6). □

Since (3.3.5) defines a particular set of choices for $S_{\mathbf{p}^{n+1}}$, the linear projection methods on moving grids constitute a subset of all possible discrete gradient methods on moving grids as defined by (3.3.1). Note also that, since the linear projection methods are independent of the discrete gradient, each linear projection method defines an equivalence class of the methods (3.3.1), uniquely defined by the choice of $g_{\mathbf{p}^{n+1}}$.

3.3.3 Family of discretized integrals

At the core of the methods considered here is the notion that an approximation to the first integral $\mathcal{I}$ is preserved, and that this approximation is dependent on the discretization parameters which may change from iteration to iteration. That is, we have a family of discretized first integrals $\mathcal{I}_{\mathbf{p}}$, and at each time step the discretized first integral is exchanged for another. For each set of discretization parameters $\mathbf{p}$, there is a corresponding set of degrees of freedom $\mathbf{u}$, in which we search for a $\mathbf{u}$ such that $\mathcal{I}_{\mathbf{p}}(\mathbf{u})$ is preserved. This can be interpreted as a fiber bundle with base space B as the set of all possible discretization parameters $\mathbf{p}$, and fibers $F_{\mathbf{p}}$ as the sets of all degrees of freedom such that the discretized first integral is equal to the initial discretized first integral, i.e. $F_{\mathbf{p}} = \{\mathbf{u} \in \mathbb{R}^M | \mathcal{I}_{\mathbf{p}}(\mathbf{u}) = \mathcal{I}_{\mathbf{p}^0}(\mathbf{u}^0)\}$. A similar idea, although without energy preservation, has been discussed by Bauer, Joshi and Modin in [2].

3.4 Numerical experiments

To provide examples of the application of our method and to investigate its accuracy, we have applied it to two one-dimensional PDEs: the sine-Gordon equation and the Korteweg–de Vries (KdV) equation. The choice of these equations were made because they both possess traveling wave solutions in the form of solitons, providing an ideal situation for r-adaptivity, which allows the grid points to cluster around wave fronts. The following experiments consider r-adaptivity only, and not p- or h-adaptivity. The sine-Gordon equation is solved using the FDM formulation of section 3.2.2, while the KdV equation is solved using the PUM formulation of section 3.2.3.

We wish to compare our methods to standard methods on fixed and adaptive meshes. This gives us four methods to consider: Fixed mesh methods with energy preservation by discrete gradients (DG), adaptive mesh methods with preservation by discrete gradients (DGMM), a non-preserving fixed grid method (MP), and the same method with adaptive mesh (MPMM). The former two methods are those described earlier in the paper, while the latter two are made differently for the two equations. In the sine-Gordon case, we use a finite difference scheme where spatial discretization is done using central finite differences and time discretization using the implicit midpoint rule. In the KdV case, the spatial discretization is performed the same way as for the discrete gradient schemes, while the time discretization is done using the implicit midpoint rule. The mesh adaptivity procedure for the DGMM and MPMM schemes is presented in the next subsection.

The MPMM scheme for the sine-Gordon equation appeared unstable unless restrictively short time steps were used, and the results of those tests are therefore omitted from the following discussion. It is difficult to analyze the MPMM scheme and pinpoint an exact cause for this instability. However, it is worth noting that the other three schemes have preservation properties that should contribute to their stability; the DG and DGMM schemes have energy preservation properties, and the semidiscretization used for the sine-Gordon equation gives rise to a Hamiltonian system of equations which means that the MP scheme, which is symplectic, should perform well. On the other hand, the moving mesh strategy used breaks the symplecticity property in the MPMM scheme; specifically, the transfer strategies as presented in the next subsection do not preserve symplecticity. The results using MPMM for the KdV equation were better, and are presented.

3.4.1 Adaptivity

Concerning adaptivity of the mesh, we used a simple method for r-adaptivity which can be applied to both FDM and FEM problems in one spatial dimension.

When applying moving mesh methods, one can either couple the evolution of the mesh with the PDE to be solved through a Moving Mesh PDE [15] or use the rezoning approach, where function values and grid points are calculated in an intermittent fashion. Since our method is based on having a new set of grid points at each time step, and not coupling the evolution of the mesh to the PDE, the latter approach was used. It is based on an equidistribution principle, meaning that when $\Omega = [a, b]$ is split into M intervals, one requires that

$$\int_{x_i}^{x_{i+1}} \omega(x)\mathrm{d}x = \frac{1}{M}\int_a^b \omega(x)\mathrm{d}x,$$

where the monitor function ω is a function measuring how densely grid points should lie, based on the value of u. The choice of monitor function is problem dependent, and choosing it optimally may require considerable research. A variety of monitor functions have been studied for certain classes of problems, see e.g. [3, 5]. Through numerical experiments, we found little difference in performance when choosing between monitor functions based on arc-length and curvature, and have in the following used the former, that is, the generalized arc-length monitor function [5]

$$\omega(x) = \sqrt{1 + k^2\left(\frac{\partial u}{\partial x}(x)\right)^2}.$$

Here, the equidistribution principle amounts to requiring that the weighted arc length (in the case $k = 1$ one recovers the usual arc length) of u over each interval is equal. In applications, we only have an approximation of u, meaning ω must be approximated as well; in our case, we have applied a finite difference approximation and obtained approximately equidistributing grids using de Boor's method as explained in [16, pp. 36-38]. We tried different smoothing techniques, including a direct smoothing of the monitor function and an iterative procedure for the regridding by De Boor's method (see e.g. [4, 16, 24]). In the case of the KdV equation, there was little to no improvement using smoothing, but the sine-Gordon experiments showed significant improvement with direct smoothing; i.e., in De Boor's algorithm, we use the smoothed discretized monitor function

$$\bar{\omega}_i = \frac{\omega_{i-1} + 2\omega_i + \omega_{i+1}}{4}.$$

Having obtained the discretization parameters for the current time step, the numerical solution $\mathbf{u}$ from the previous time step must be transferred onto the new set of mesh points. We tested three different ways of doing this, two of which are using linear interpolation and cubic interpolation. The linear

interpolation consists of constructing a function $\hat{u}(x)$ which is piecewise linear on each interval $[x_i^n, x_{i+1}^n]$ such that $\hat{u}(x_i^n) = u_i^n$, then evaluating this function at the new mesh points, giving the interpolated values $\hat{u}_i = \hat{u}(x_i^{n+1})$. The cubic interpolation consists of a similar construction, using cubic Hermite splines through the MATLAB function `pchip`. Of these two transfer methods, the cubic interpolation yielded superior results in all cases, and so only results using cubic interpolation are presented. The third way, using preserving transfer as presented in section 3.3.1, applies to the KdV example, where the PUM is used. Here, we found little difference between cubic interpolation and exact transfer, so results are presented using cubic interpolation for the transfer operation here as well.

3.4.2 Sine-Gordon equation

The sine-Gordon equation is a nonlinear hyperbolic PDE in one spatial and one temporal dimension exhibiting soliton solutions, with applications in predicting dislocations in crystals and propagation of fluxons in junctions between superconductors. It is stated in initial value problem form as:

$$\begin{aligned} &u_{tt} - u_{xx} + \sin(u) = 0, \quad (x,t) \in \mathbb{R} \times [0,T], \\ &u(x,0) = f(x), \quad u_t(x,0) = g(x). \end{aligned} \tag{3.4.1}$$

We consider a finite domain $[-L, L] \times [0, T]$ with periodic boundary conditions $u(-L) = u(L)$ and $u_t(-L) = u_t(L)$. The equation has the first integral

$$\mathcal{I}[u] = \int_{\mathbb{R}} \frac{1}{2}u_t^2 + \frac{1}{2}u_x^2 + 1 - \cos(u)\mathrm{d}x.$$

Introducing $v = u_t$, (3.4.1) can be rewritten as a first-order system of PDEs:

$$\begin{bmatrix} u_t \\ v_t \end{bmatrix} = \begin{bmatrix} v \\ u_{xx} - \sin(u) \end{bmatrix},$$

with first integral

$$\mathcal{I}[u,v] = \int_{\mathbb{R}} \frac{1}{2}v^2 + \frac{1}{2}u_x^2 + 1 - \cos(u)\mathrm{d}x. \tag{3.4.2}$$

Finding the variational derivative of this, one can interpret the equation in the form (3.2.3) with S and $\frac{\delta \mathcal{I}}{\delta u}$ as follows:

$$S = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}, \quad \frac{\delta \mathcal{I}}{\delta u}[u,v] = \begin{bmatrix} \sin(u) - u_{xx} \\ v \end{bmatrix}.$$

We will apply the FDM approach presented in section 3.2.2, approximating (3.4.2) by some quadrature with points $\{x_i\}_{i=0}^M$ and weights $\{\kappa_i\}_{i=0}^M$,

$$\mathcal{I}[u,v] \simeq \sum_{i=0}^{M} \kappa_i \left(\frac{1}{2} v_i^2 + \frac{1}{2} u_{x,i}^2 + 1 - \cos(u_i) \right).$$

In addition, we approximate the spatial derivatives with central differences. At the endpoints, a periodic extension is assumed, yielding the approximation

$$\mathcal{I}_{\mathbf{p}}(\mathbf{u}) = \sum_{i=0}^{M} \kappa_i \left(\frac{1}{2} v_i^2 + \frac{1}{2} \left(\frac{\delta u_i}{\delta x_i} \right)^2 + 1 - \cos(u_i) \right).$$

Here, $\delta w_i = w_{i+1} - w_{i-1}$ denotes central difference, with special cases $\delta u_0 = \delta u_M = u_1 - u_{M-1}$, and $\delta x_0 = \delta x_M = x_1 - x_0 + x_M - x_{M-1}$. Taking the gradient of $\mathcal{I}_{\mathbf{p}}(\mathbf{u})$ and applying the AVF discrete gradient gives

$$\overline{\nabla} \mathcal{I}_{\mathbf{p}}(\mathbf{u}^n, \mathbf{u}^{n+1}) = \int_0^1 \nabla_{\mathbf{u}} \mathcal{I}_{\mathbf{p}}(\xi \mathbf{u}^n + (1-\xi)\mathbf{u}^{n+1}) \mathrm{d}\xi$$

The periodic boundary conditions are enforced by setting $u_0 = u_M$. In the implementation, the κ_i were chosen as the quadrature weights associated with the composite trapezoidal rule, i.e.

$$\kappa_0 = \frac{x_1 - x_0}{2}, \quad \kappa_M = \frac{x_M - x_{M-1}}{2}, \quad \kappa_i = \frac{x_{i+1} - x_{i-1}}{2}, \quad i = 1, ..., M-1.$$

Furthermore, S was approximated by the matrix

$$S_d = \begin{bmatrix} 0 & I \\ -I & 0 \end{bmatrix},$$

with I an $M \times M$ identity matrix. The exact solution considered was

$$u(x,t) = 4 \tan^{-1} \left(\frac{\sinh\left(\frac{ct}{\sqrt{1-c^2}} \right)}{c \cosh\left(\frac{x}{\sqrt{1-c^2}} \right)} \right).$$

This is a *kink-antikink* system, an interaction between two solitons, each moving in different directions with speed $c \in (0,1)$, resulting in two wave fronts traveling in opposite directions. The wave fronts become steeper as $c \to 1$. Figure 3.1 illustrates the analytical solution and shows the time evolution of the

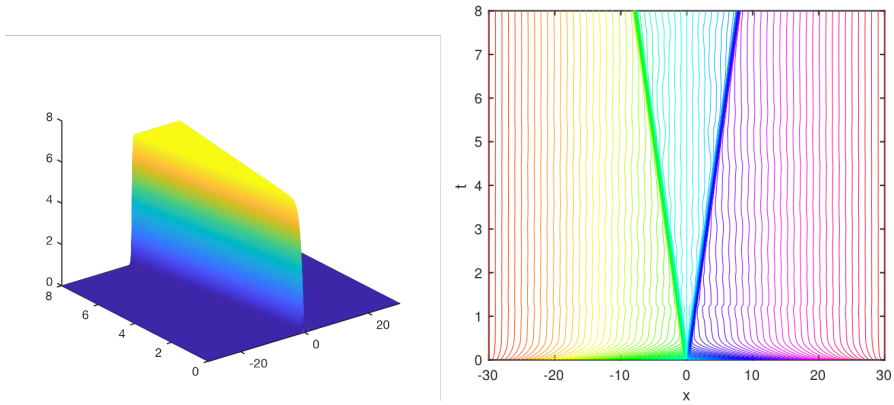


Figure 3.1: *Left*: Illustration of kink-antikink solution. *Right*: Grid movement - each line represents the path of one grid point in time.

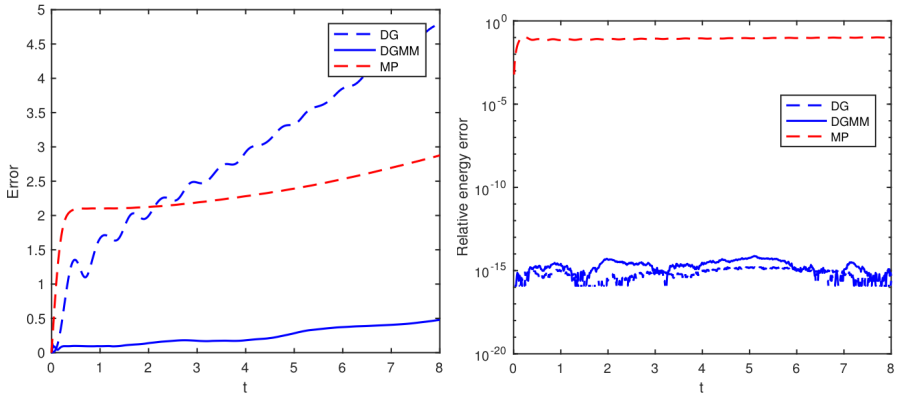


Figure 3.2: *Left*: L_2 error. *Right*: Relative error in $I_{\mathbf{p}}$. *Parameters*: $\Delta t = 0.01$, $M = 300$, $L = 30$, $c = 0.99$.

mesh as obtained with the DGMM method. Note that the grid points cluster along the wave fronts.

The left hand side of Figure 3.2 shows the time evolution of the error $E_n^u = ||u_n^I(x) - u(x,t_n)||_{L_2}$, where u_n^I is a linear interpolant created from the pairs $(\mathbf{u}^n, \mathbf{x}^n)$. The right hand side of Figure 3.2 shows the time evolution of the relative error in the discretized energy, $E_n^I = (I_{\mathbf{p}^n}(\mathbf{u}^n) - I_{\mathbf{p}^0}(\mathbf{u}^0))/I_{\mathbf{p}^0}(\mathbf{u}^0)$. We can see that the long-term behaviour of the MP scheme is superior to that of the DG scheme, but when mesh adaptivity is applied, the DGMM scheme is clearly better. Also note that while the DG and DGMM schemes preserve $I_{\mathbf{p}}$ to machine precision, the MP scheme does not.

Figure 3.3 shows the convergence behaviour of the three schemes with

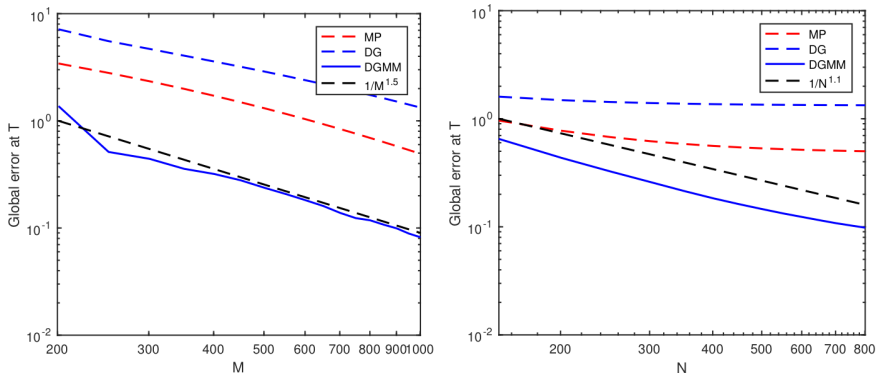


Figure 3.3: *Left*: Error at $T = 8$ as a function of M, with $\Delta t = 0.008$, $c = 0.99$, $L = 30$. *Right*: Error at $T = 8$ as a function of $N = T/\Delta t$, with $M = 1000$, $c = 0.99$, $L = 30$.

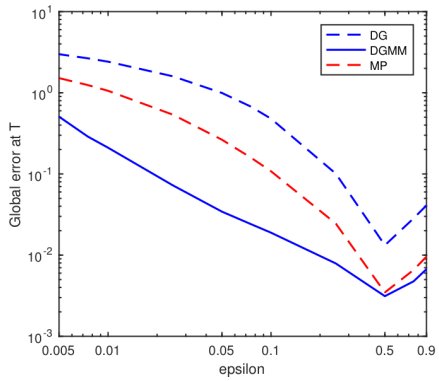


Figure 3.4: Error at $T = 8$ as a function of ε, with $\Delta t = 0.01$, $M = 600$ and $L = 30$.

respect to the number of spatial discretization points M, and the number of time steps N. Note that the DG and MP methods plateau at $N \simeq 400$; this is due to the error stemming from spatial discretization dominating the time discretization error for these methods, while the DGMM scheme has lower spatial discretization error. The convergence order of the DGMM scheme was measured using a first order polynomial fitting of $\log(E_n^u)$ to $\log(M)$ and $\log(N)$. The convergence order with respect to M was calculated as 1.518, and the convergence order with respect to N was measured at 1.121.

Finally, to illustrate the applicability of the DGMM scheme to harder problems, Figure 3.4 shows the error at stopping time of the methods as a function of a parameter ϵ representing the increasing speed of the solitons ($c = 1 - \varepsilon$). From this plot, it is appararent that while the non-adaptive MP scheme is competitive at low speeds, the moving mesh method provides significantly more accuracy as $c \to 1$.

3.4.3 Korteweg–de Vries equation

The KdV equation is a nonlinear PDE with soliton solutions modelling shallow water surfaces, stated as

$$u_t + u_{xxx} + 6uu_x = 0. \tag{3.4.3}$$

It has infinitely many first integrals, one of which is the Hamiltonian

$$\mathcal{H}[u] = \int_{\mathbb{R}} \frac{1}{2}u_x^2 - u^3 \mathrm{d}x.$$

With this Hamiltonian, we can write (3.4.3) in the form (3.2.3) with S and $\frac{\delta \mathcal{H}}{\delta u}$ as follows:

$$S = \frac{\partial}{\partial x}, \quad \frac{\delta \mathcal{H}}{\delta u}[u] = -u_{xx} - 3u^2.$$

We will apply the PUM approach to create a numerical scheme which preserves an approximation to $\mathcal{H}[u]$, splitting $\Omega = [-L, L]$ into M elements $\{[x_i, x_{i+1}]\}_{i=0}^{M-1}$ and using Lagrangian basis functions φ_j of arbitrary degree for the trial space. Approximating u by u^h as in section 3.2.3, we find

$$\begin{aligned}\mathcal{H}_{\mathbf{p}}(\mathbf{u}) = \mathcal{H}[u^h] &= \int_\Omega \frac{1}{2}(u_x^h)^2 - (u^h)^3 \mathrm{d}x \\ &= \frac{1}{2}\sum_{j,k} u_j u_k \int_\Omega \varphi_{j,x}\varphi_{k,x}\mathrm{d}x - \sum_{j,k,l} u_j u_k u_l \int_\Omega \varphi_j \varphi_k \varphi_l \mathrm{d}x. \end{aligned} \tag{3.4.4}$$

The integrals can be evaluated exactly and efficiently by considering element-wise which basis functions are supported on the element before applying Gaussian quadrature to obtain exact evaluations of the polynomial integrals. We define

$$D_{ijk} = \int_\Omega \varphi_i \varphi_j \varphi_k \mathrm{d}x \quad \text{and} \quad E_{ij} = \int_\Omega \varphi_{i,x}\varphi_{j,x}\mathrm{d}x.$$

The matrices A and B with

$$A_{ij} = \int_\Omega \varphi_i \varphi_j \mathrm{d}x \quad \text{and} \quad B_{ji} = \int_\Omega \varphi_i \varphi_{j,x}\mathrm{d}x$$

are formed in the same manner. Note that B is in this case independent of $\mathbf{u}$. Applying the AVF method yields the discrete gradient

$$\overline{\nabla}\mathcal{H}_{\mathbf{p}}(\mathbf{u}^n, \mathbf{u}^{n+1}) = \int_0^1 \nabla_{\mathbf{u}}\mathcal{H}_{\mathbf{p}}(\xi\mathbf{u}^n + (1-\xi)\mathbf{u}^{n+1})\mathrm{d}\xi$$

such that, with the convention of summation over repeated indices,

$$(\overline{\nabla}\mathcal{H}_{\mathbf{p}})_i = \frac{1}{2}E_{ij}(u_j^n + u_j^{n+1}) - D_{ijk}(u_j^n(u_k^n + \frac{1}{2}u_k^{n+1}) + u_j^{n+1}(\frac{1}{2}u_k^n + u_k^{n+1})).$$

This gives us all the required terms for forming the system (3.2.15) and applying the discrete gradient method to it. During testing, the φ_j were chosen as piecewise linear polynomials. The exact solution considered is of the form

$$u(x,t) = \frac{c}{2}\text{sech}^2\left(\frac{\sqrt{c}}{2}(x - ct)\right), \tag{3.4.5}$$

which is a right-moving soliton with c as the propagation speed, chosen as $c = 6$ in the numerical tests. We have considered periodic boundary conditions on a domain $[-L, L] \times [0, T]$, with $L = 100$ in all the following results.

Our discrete gradient method on a moving mesh (DGMM) is compared to the same method on a static, equidistributed mesh (DG), and the implicit midpoint method on static (MP) and moving mesh (MPMM). The spatial discretization is performed the same way in all cases. Figure 3.5 shows an example of exact and numerical solutions at $t = 15$. Note that the peak in the exact solution will be located at $x = ct$.

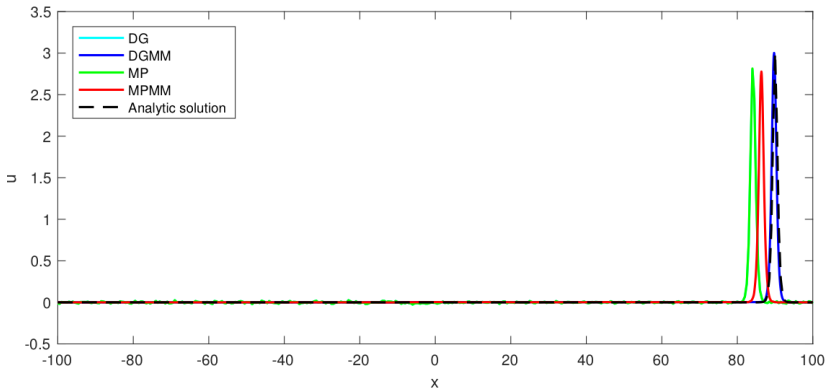


Figure 3.5: Solutions at T = 15. $\Delta t = 0.01$, $M = 400$. MP and DG are almost indistinguishable.

To evaluate the numerical solution, it is reasonable to look at the distance error

$$E_n^{\text{dist}} = ct_n - x^*,$$

where $x^* = \arg\max_x u_h(x, t_n)$, i.e. the location of the peak in the numerical solution. Another measure of the error is the shape error

$$E_n^{\text{shape}} = \left|\left| u_h(x, t_n) - u\left(x, \frac{x^*}{c}\right)\right|\right|,$$

where the peak of the exact solution is translated to match the peak of the numerical solution, and the shapes of the solitons are compared.

Figure 3.6 confirms that the DG and DGMM methods preserve the approximated Hamiltonian (3.4.4), while it is also worth noting that in the case of the midpoint method, the error in this conserved quantity is much larger on a moving than on a static mesh. Similar behaviour is also observed for a moving-mesh

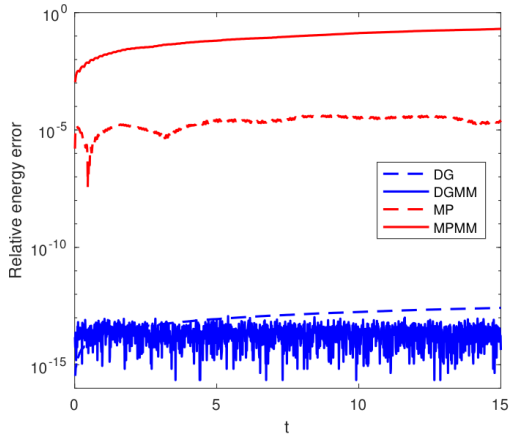


Figure 3.6: Relative error in the Hamiltonian plotted as a function of time $t \in [0, 15]$. $\Delta t = 0.01$, $M = 400$.

method for the regularized long wave equation in the recent paper [19], where it is concluded that a moving mesh method with a conservative property would be an interesting research topic. Figure 3.7, where the phase and shape errors are plotted up to $T = 15$, is an example of how the DGMM method performs comparatively better with increasing time.

In figures 3.8 and 3.9 we present the phase and shape errors for the different methods as a function of the number of elements M and the number of time steps N, respectively. Reference lines are included to give an indication of the rate of convergence. We also calculated this for the DGMM method by first degree polynomial fitting of the error curve, giving a convergence order of 1.135 for the phase error and 2.311 for the shape error as a function of M. As a function of N, we get a convergence order of 1.492 for the phase error, and 1.609 for the shape error (the latter measured up to $N = 320$, where it flattens out). We observe that the DGMM scheme performs especially well, compared to the other three schemes, for a coarse spatial discretization compared to the discretization in time.

In figure 3.10, the phase and shape errors are plotted as a function of the parameter c in the exact solution (3.4.5), where we note that $\frac{c}{2}$ is the height of the wave; increasing c leads to sharper peaks and thus a harder numerical problem. As expected, the advantages of the DGMM method is less evident

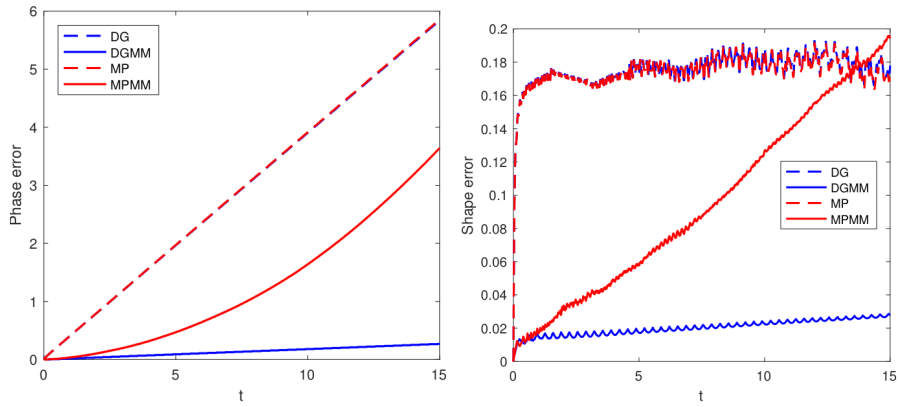


Figure 3.7: Phase error (left) and shape error (right) as a function of time. $\Delta t = 0.01$, $M = 400$.

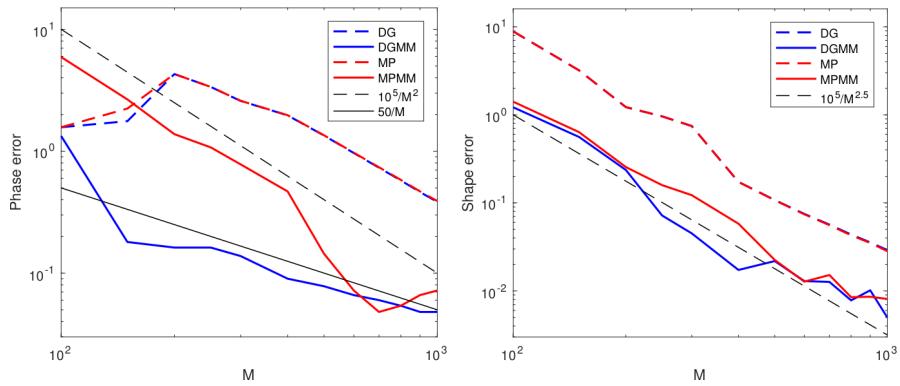


Figure 3.8: Phase error (left) and shape error (right) as a function of the number of elements M, at time $T = 5$. $\Delta t = 0.01$.

for small c, but we observe that the DGMM method outperforms the static grid midpoint method already when $c = 2$.

3.4.4 Execution time

The code used is not optimized, so any quantitative comparison to standard methods has not been performed; it is still possible to make some qualitative observations. Adding adaptivity increases time per iteration slightly since the systems become more complicated, especially in the case of the PUM approach where the matrices A and B need to be recalculated, at each time step when adaptivity is used. This increases runtime somewhat when compared to fixed grid methods. However, adaptivity allows for using fewer degrees of freedom,

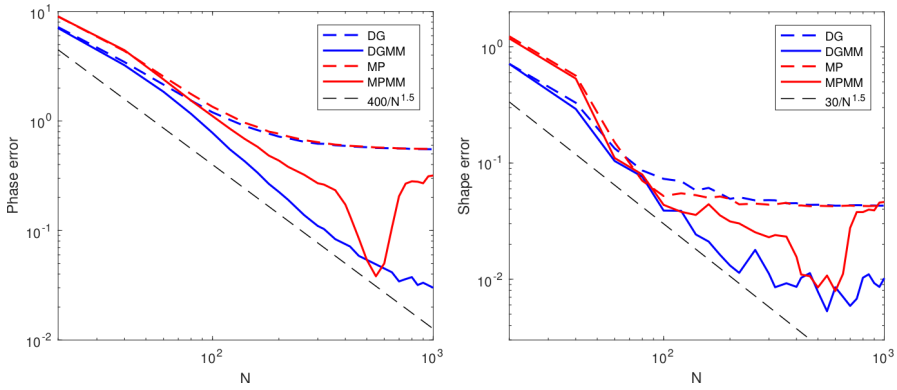


Figure 3.9: Phase error (left) and shape error (right) at time $T = 5$, as a function of the number of time steps $N = T/\Delta t$. $M = 800$.

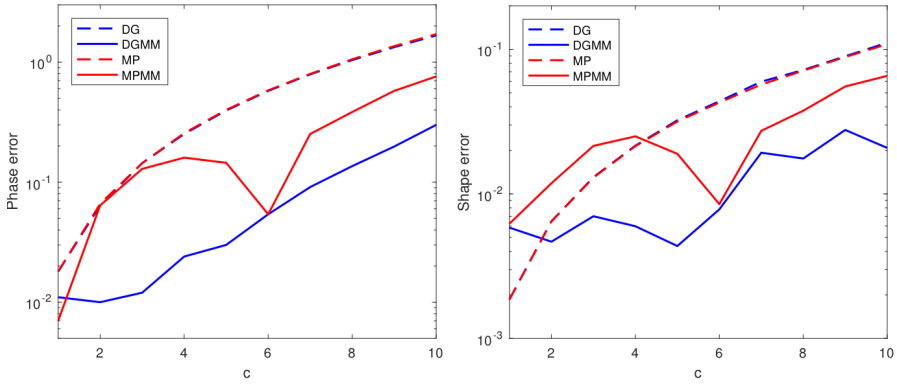


Figure 3.10: Phase error (left) and shape error (right) as a function of c in the exact solution (3.4.5), at time $t = 5$. $\Delta t = 0.01$, $M = 800$.

and so decreases the degrees of freedom needed for a given level of accuracy. This accuracy gain is more pronounced the harder the problem is (steeper wave fronts etc.), and so it stands to reason that there will be situations where adaptive energy preserving methods will outperform non-adaptive and/or non-preserving methods.

3.5 Conclusion

In this paper, we have introduced a general framework for producing adaptive first integral preserving methods for partial differential equations. This is done by first providing two means of producing first integral preserving methods on arbitrary fixed grids, then showing how to extend these methods to allow

for adaptivity while preserving the first integral. Numerical testing shows that moving mesh methods coupled with discrete gradient methods provide good solvers for the sine-Gordon and Korteweg–de Vries equations. It would be of interest to apply the method to higher-dimensional PDEs with a more challenging geometry, preferably using the PUM approach, to investigate its accuracy as compared to conventional methods, and to test whether *h*- and/or *p*-refinement provides a notable improvement. It may also prove fruitful to explore the ideas presented in [2] to make the transfer operations between sets of discretization parameters in a more natural setting than simply interpolating, as suggested in section 3.3.3. Furthermore, analysis of the methods considered here could provide important insight into e.g. stability, consistency and convergence order.

Bibliography

[1] I. BABUŠKA AND B. GUO, *The h, p and h-p version of the finite element method; basis theory and applications*, Adv. Eng. Softw., 15 (1992), pp. 159–174.

[2] M. BAUER, S. JOSHI, AND K. MODIN, *Diffeomorphic density matching by optimal information transport*, SIAM J. Imaging Sci., 8 (2015), pp. 1718–1751.

[3] J. BLOM AND J. VERWER, *On the use of the arclength and curvature monitor in a moving-grid method which is based on the method of lines*, tech. rep., NM-N8902, CWI, Amsterdam, 1989.

[4] C. J. BUDD, W. HUANG, AND R. D. RUSSELL, *Moving mesh methods for problems with blow-up*, SIAM J. Sci. Comput., 17 (1996), pp. 305–327.

[5] C. J. BUDD, W. HUANG, AND R. D. RUSSELL, *Adaptivity with moving grids*, Acta Numer., 18 (2009), pp. 111–241.

[6] E. CELLEDONI, V. GRIMM, R. I. MCLACHLAN, D. I. MCLAREN, D. O'NEALE, B. OWREN, AND G. R. W. QUISPEL, *Preserving energy resp. dissipation in numerical PDEs using the "average vector field" method*, J. Comput. Phys., 231 (2012), pp. 6770–6789.

[7] S. H. CHRISTIANSEN, H. Z. MUNTHE-KAAS, AND B. OWREN, *Topics in structure-preserving discretization*, Acta Numer., 20 (2011), pp. 1–119.

[8] R. COURANT, K. FRIEDRICHS, AND H. LEWY, *Über die partiellen Differenzengleichungen der mathematischen Physik*, Math. Ann., 100 (1928), pp. 32–74.

[9] M. DAHLBY AND B. OWREN, *A general framework for deriving integral preserving numerical methods for PDEs*, SIAM J. Sci. Comput., 33 (2011), pp. 2318–2340.

[10] D. FURIHATA, *Finite difference schemes for $\partial u/\partial t = (\partial/\partial x)^{\alpha}\delta G/\delta u$ that inherit energy conservation or dissipation property*, J. Comput. Phys., 156 (1999), pp. 181–205.

[11] D. FURIHATA, *Finite-difference schemes for nonlinear wave equation that inherit energy conservation property*, J. Comput. Appl. Math., 134 (2001), pp. 37–57.

[12] D. FURIHATA AND T. MATSUO, *Discrete variational derivative method*, Chapman & Hall/CRC Numerical Analysis and Scientific Computing, CRC Press, Boca Raton, FL, 2011.

[13] O. GONZALEZ, *Time integration and discrete Hamiltonian systems*, J. Nonlinear Sci., 6 (1996), pp. 449–467.

[14] E. HAIRER, C. LUBICH, AND G. WANNER, *Geometric numerical integration*, vol. 31 of Springer Series in Computational Mathematics, Springer, Heidelberg, 2010.

[15] W. HUANG AND R. RUSSELL, *Adaptive mesh movement - the MMPDE approach and its applications*, J. Comput. Appl. Math., 128 (2001), pp. 383–398.

[16] W. HUANG AND R. RUSSELL, *Adaptive moving mesh methods*, vol. 174 of Springer Series in Applied Mathematical Sciences, Springer-Verlag, New York, 2010.

[17] T. LEE, M. BAINES, AND S. LANGDON, *A finite difference moving mesh method based on conservation for moving boundary problems*, J. Comput. Appl. Math., 288 (2015), pp. 1–17.

[18] S. LI AND L. VU-QUOC, *Finite difference calculus invariant structure of a class of algorithms for the nonlinear Klein-Gordon equation*, SIAM J. Numer. Anal., 32 (1995), pp. 1839–1875.

[19] C. LU, W. HUANG, AND J. QIU, *An adaptive moving mesh finite element solution of the regularized long wave equation*, J. Sci. Comput., 74 (2018), pp. 122–144.

[20] R. I. MCLACHLAN, G. R. W. QUISPEL, AND N. ROBIDOUX, *Geometric integration using discrete gradients*, Philos. T. R. Soc. A, 357 (1999), pp. 1021–1045.

[21] J. MELENK AND I. BABUŠKA, *The partition of unity finite element method: Basic theory and applications*, Comput. Method. Appl. M., 139 (1996), pp. 289–314.

[22] Y. MIYATAKE AND T. MATSUO, *A note on the adaptive conservative/dissipative discretization for evolutionary partial differential equations*, J. Comput. Appl. Math., 274 (2015), pp. 79–87.

[23] R. A. NORTON, D. I. MCLAREN, G. R. W. QUISPEL, A. STERN, AND A. ZANNA, *Projection methods and discrete gradient methods for preserving first integrals of ODEs*, Discrete Contin. Dyn. Syst., 35 (2015), pp. 2079–2098.

[24] J. D. PRYCE, *On the convergence of iterated remeshing*, IMA J. Numer. Anal., 9 (1989), pp. 315–335.

[25] G. R. W. QUISPEL AND D. I. MCLAREN, *A new class of energy-preserving numerical integration methods*, J. Phys. A: Math. Theor., 41 (2008).

[26] R. D. RICHTMYER AND K. W. MORTON, *Difference methods for initial-value problems*, Second edition. Interscience Tracts in Pure and Applied Mathematics, No. 4, Interscience Publishers John Wiley & Sons, Inc., New York-London-Sydney, 1967.

[27] T. YAGUCHI, T. MATSUO, AND M. SUGIHARA, *An extension of the discrete variational method to nonuniform grids*, J. Comput. Phys., 229 (2010), pp. 4382–4423.

[28] T. YAGUCHI, T. MATSUO, AND M. SUGIHARA, *The discrete variational derivative method based on discrete differential forms*, J. Comput. Phys., 231 (2012), pp. 3963–3986.

[29] P. A. ZEGELING, *r-refinement for evolutionary PDEs with finite elements or finite differences*, Appl. Numer. Math., 26 (1998), pp. 97–104.

Energy preserving moving mesh methods applied to the BBM equation

Sølve Eidnes and Torbjørn Ringholm

Published in Proceedings of MekIT '17

Energy preserving moving mesh methods applied to the BBM equation

Abstract. Energy preserving numerical methods for a certain class of PDEs are derived, applying the partition of unity method. The methods are extended to also be applicable in combination with moving mesh methods by the rezoning approach. These energy preserving moving mesh methods are then applied to the Benjamin–Bona–Mahony equation, resulting in schemes that exactly preserve an approximation to one of the Hamiltonians of the system. Numerical experiments that demonstrate the advantages of the methods are presented.

4.1 Introduction

Numerical solutions of differential equations by discretization methods will typically not inherit invariant properties from the continuous problem. Since the energy preserving methods of Courant, Friedrichs and Lewy were introduced in [8], conservative methods have inspired much research, surveyed in [15] up to the early 1990s. In some cases, conservation properties can ensure numerical stability or existence and uniqueness of the numerical solution. In other cases, conservation of one or more invariants can be of importance in its own right. As noted in [13], one may expect that when properties of a continuous dynamical system are inherited by the discrete dynamical system, the numerical solution will be more accurate, especially over large time intervals.

The discrete gradient methods for ordinary differential equations (ODEs), usually attributed to Gonzalez [12], are methods that preserve first integrals exactly. Since the late 1990s, a number of researchers have worked on extending this theory to create a counterpart for partial differential equations (PDEs), see e.g. [5, 11]. Such methods, called either discrete variational derivative methods or discrete gradient methods for PDEs, preserve a discrete approximation of a first integral. Until recently, the schemes presented have been based on a finite difference approach, and exclusively on fixed, uniform grids. Two discrete variational derivative methods on fixed, non-uniform grids were presented by Yaguchi, Matsuo and Sugihara in [21, 22]. In [18], Miyatake and Matsuo introduce integral preserving methods on adaptive grids for certain classes of PDEs. Eidnes, Owren and Ringholm presented in [10] a general approach to extending the theory of discrete variational derivative methods, or discrete gradient methods for PDEs, to adaptive grids. This is done using either finite differences or the partition of unity method, a generalization of the finite element method.

In this paper, we present an application of the approach introduced in [10] to the Benjamin–Bona–Mahony (BBM) equation, also called the regularized

long wave equation in the literature. Although what we present here is a finite element method, the theory can be easily applied in a finite difference setting. Previously, there have been developed integral preserving methods for this equation [6], as well as adaptive moving mesh methods [16], but the schemes we are to present here are, to our knowledge, the first combining these properties. In fact, in [16] it is noted that combining integral preservation with adaptivity is an interesting topic for further research.

4.2 The discrete gradient method for PDEs

We give a quick survey of the discrete gradient methods for PDEs, and present an approach to the spatial discretization by the partition of unity method (PUM).

4.2.1 Problem statement

Consider a PDE of the form

$$u_t = f(\mathbf{x}, u^J), \qquad \mathbf{x} \in \Omega \subseteq \mathbb{R}^d, \quad u \in \mathcal{B} \subseteq L^2, \tag{4.2.1}$$

where u^J denotes u itself and its partial derivatives of any order with respect to the spatial variables $x_1, ..., x_d$, and where we assume that $\mathcal{B}$ is sufficiently regular to allow all operations used in the following.

We define a *first integral* of (4.2.1) to be a functional $\mathcal{I}[u]$ satisfying

$$\left\langle \frac{\delta \mathcal{I}}{\delta u}[u], f(\mathbf{x}, u^J) \right\rangle_{L^2} = 0, \quad \forall u \in \mathcal{B},$$

recalling that the *variational derivative* $\frac{\delta \mathcal{I}}{\delta u}[u]$ is defined as the function satisfying

$$\left\langle \frac{\delta \mathcal{I}}{\delta u}[u], v \right\rangle_{L^2} = \frac{\mathrm{d}}{\mathrm{d}\epsilon}\bigg|_{\epsilon=0} \mathcal{I}[u + \epsilon v] \quad \forall v \in \mathcal{B}.$$

This means that $\mathcal{I}[u]$ is preserved over time by (4.2.1), since

$$\frac{\mathrm{d}\mathcal{I}}{\mathrm{d}t} = \left\langle \frac{\delta \mathcal{I}}{\delta u}[u], \frac{\partial u}{\partial t} \right\rangle_{L^2} = 0.$$

Furthermore, observe that if there exists an operator $S(\mathbf{x}, u^J)$, skew-symmetric with respect to the L^2 inner product, such that

$$f(\mathbf{x}, u^J) = S(\mathbf{x}, u^J) \frac{\delta \mathcal{I}}{\delta u}[u],$$

then $\mathcal{I}[u]$ is a first integral of (4.2.1), and we can state (4.2.1) on the form

$$u_t = S(\mathbf{x}, u^J)\frac{\delta \mathcal{I}}{\delta u}[u]. \tag{4.2.2}$$

The idea behind the discrete variational derivative methods is to derive a discrete version of the PDE on the form (4.2.2), by obtaining a so-called discrete variational derivative and approximate $S(\mathbf{x}, u^J)$ by a skew-symmetric matrix, see e.g. [11].

As proven in [10], all discrete variatonal derivative methods can be expressed as discrete gradient methods on a system of ODEs obtained by discretizing (4.2.2) in space, to get a system

$$\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} = S(\mathbf{u})\nabla I(\mathbf{u}), \tag{4.2.3}$$

where $S(\mathbf{u})$ is a skew-symmetric matrix. The discrete gradient methods for such a system of ODEs preserve the first integral $I(\mathbf{u})$ [17]. These numerical methods are given by

$$\frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\Delta t} = \bar{S}(\mathbf{u}^n, \mathbf{u}^{n+1})\overline{\nabla} I(\mathbf{u}^n, \mathbf{u}^{n+1}),$$

where $\bar{S}(\mathbf{u}^n, \mathbf{u}^{n+1})$ is a consistent skew-symmetric time-discrete approximation to $S(\mathbf{u})$ and $\overline{\nabla} I(\mathbf{v}, \mathbf{u})$ is a discrete gradient of $I(\mathbf{u})$, defined as a function satisfying

$$(\overline{\nabla} I(\mathbf{v}, \mathbf{u}))^T(\mathbf{u} - \mathbf{v}) = I(\mathbf{u}) - I(\mathbf{v}),$$
$$\overline{\nabla} I(\mathbf{u}, \mathbf{u}) = \nabla I(\mathbf{u}).$$

There are many possible choices of discrete gradients. For the numerical experiments in this note, we will use the Average Vector Field (AVF) discrete gradient [5], given by

$$\overline{\nabla} I(\mathbf{v}, \mathbf{u}) = \int_0^1 \nabla I(\xi\mathbf{u} + (1-\xi)\mathbf{v})\mathrm{d}\xi,$$

Note that when discretizing the system (4.2.2) in space, we do so by finding a discrete approximation $\mathcal{I}_{\mathbf{p}}$ to the integral $\mathcal{I}$, and define an energy preserving method to be a method preserving this approximation.

4.2.2 Partition of unity method on a fixed mesh

The partition of unity method is a generalization of the finite element method (FEM). Stating a weak form of (4.2.2), the problem consists of finding $u \in \mathcal{B}$

such that

$$\langle u_t, v\rangle_{L^2} = \left\langle S(\mathbf{x}, u^J)\frac{\delta \mathcal{I}}{\delta u}[u], v \right\rangle_{L^2} = -\left\langle \frac{\delta \mathcal{I}}{\delta u}[u], S(\mathbf{x}, u^J)v \right\rangle_{L^2} \qquad \forall v \in \mathcal{B}.$$

We define an approximation to u by

$$u^h(x,t) = \sum_{i=0}^{M} u_i(t)\varphi_i(x),$$

where the test functions $\varphi_i(x)$ span a finite-dimensional subspace $\mathcal{B}^h \subseteq \mathcal{B}$. Referring to [10] for details, we then obtain the Galerkin form of the problem: Find $u_i(t), i = 0, \dots, M$, such that

$$\sum_{i=0}^{M} \frac{\mathrm{d}u_i}{\mathrm{d}t}\left\langle \varphi_i, \varphi_j \right\rangle_{L^2} = -\sum_{i=0}^{M} w_i(\mathbf{u})\left\langle \varphi_i, S(\mathbf{x}, u^{h,J})\varphi_j \right\rangle_{L^2} \qquad \forall j \in \{0, ..., M\},$$

where, with $A_{ij} = \left\langle \varphi_i, \varphi_j \right\rangle_{L^2}$,

$$\mathbf{w}(\mathbf{u}) = A^{-1}\nabla \mathcal{I}_{\mathbf{p}}(\mathbf{u}).$$

We end up with an ODE for the coefficients u_i:

$$\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} = S_{\mathbf{p}}(\mathbf{u})\nabla \mathcal{I}_{\mathbf{p}}(\mathbf{u}). \tag{4.2.4}$$

Here, $S_{\mathbf{p}}(\mathbf{u}) = -A^{-1}B(\mathbf{u})A^{-1}$ is a skew-symmetric matrix, with $B(\mathbf{u})$ given by $B(\mathbf{u})_{ji} = \left\langle \varphi_i, S(\mathbf{x}, u^{h,J})\varphi_j \right\rangle_{L^2}$, and the system is thereby of the form (4.2.3). Then, the scheme

$$\frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\Delta t} = S_{\mathbf{p}}(\mathbf{u}^n, \mathbf{u}^{n+1})\overline{\nabla}\mathcal{I}_{\mathbf{p}}(\mathbf{u}^n, \mathbf{u}^{n+1}).$$

will preserve the approximated first integral $\mathcal{I}_{\mathbf{p}}$ in the sense that $\mathcal{I}_{\mathbf{p}}(\mathbf{u}^{n+1}) = \mathcal{I}_{\mathbf{p}}(\mathbf{u}^n)$.

4.3 Adaptive schemes

The primary motivation for using an adaptive mesh is usually to increase accuracy while keeping computational cost low, by improving discretization locally. Such methods are typically useful for problems with e.g. traveling wave solutions and boundary layers. The different strategies for adaptive meshes can be classified into two main groups [14]: The quasi-Lagrange approach involves coupling the evolution of the mesh with the PDE, and then solving the problems simultaneously; The rezoning approach consists of calculating the function values and mesh points in an intermittent fashion. Our method can be coupled with any adaptive mesh strategy utilizing the latter approach.

4.3.1 Adaptive discrete gradient methods

Let $\mathbf{p}^n$, $\mathbf{u}^n$, $\mathbf{p}^{n+1}$, and $\mathbf{u}^{n+1}$ denote the discretization parameters and the numerical values obtained at the current time step and next time step, respectively. Note that we now alter the notion of a preserved first integral further, to requiring that $\mathcal{I}_{\mathbf{p}^{n+1}}(\mathbf{u}^{n+1}) = \mathcal{I}_{\mathbf{p}^n}(\mathbf{u}^n)$. The idea behind our approach is to find $\mathbf{p}^{n+1}$ based on $\mathbf{u}^n$ and $\mathbf{p}^n$, transfer $\mathbf{u}^n$ to $\mathbf{p}^{n+1}$ to obtain $\hat{\mathbf{u}}$, and then use $\hat{\mathbf{u}}$ to propagate in time to get $\mathbf{u}^{n+1}$. If the transfer operation between the meshes is preserving, i.e. if $\mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}) = \mathcal{I}_{\mathbf{p}^n}(\mathbf{u}^n)$, then the next time step can be taken with the discrete gradient method for static meshes. If, however, non-preserving transfer is used, corrections are needed in order to get a numerical scheme. We introduce in [10] the scheme

$$\mathbf{u}^{n+1} = \hat{\mathbf{u}} - \frac{(\mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}) - \mathcal{I}_{\mathbf{p}^n}(\mathbf{u}^n))\mathbf{z}}{\left\langle \overline{\nabla}\mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}, \mathbf{u}^{n+1}), \mathbf{z} \right\rangle} + \Delta t S_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}, \mathbf{u}^{n+1}) \overline{\nabla}\mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}, \mathbf{u}^{n+1}), \quad (4.3.1)$$

where $\mathbf{z}$ is a vector which should be chosen so as to obtain a minimal correction, and such that $\langle \overline{\nabla}\mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}, \mathbf{u}^{n+1}), \mathbf{z} \rangle \neq 0$. In the numerical experiments to follow, we have used $\mathbf{z} = \overline{\nabla}\mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}, \mathbf{u}^{n+1})$.

A preserving transfer can by obtained using the method of Lagrange multipliers. Depending on whether r- p- or h-refinement (or a combination) is used between time steps, we expect the shape and/or number of basis functions to change. See e.g. [14] or [1] for examples of how the basis may change through adaptivity. Denote by $\mathcal{B}^h = \text{span}\{\varphi_i\}_{i=0}^M$ the trial space from the current time step and by $\hat{\mathcal{B}}^h = \text{span}\{\hat{\varphi}_i\}_{i=0}^{\hat{M}}$ the trial space for the next time step, and note that in general, $M \neq \hat{M}$. We wish to transfer the approximation u^h from $\mathcal{B}^h$ to $\hat{\mathcal{B}}^h$, obtaining an approximation $\hat{u}^h$, while conserving the first integral, i.e. $\mathcal{I}[u^h] = \mathcal{I}[\hat{u}^h]$. This can be formulated as a constrained minimization problem:

$$\min_{\hat{u}^h \in \tilde{\mathcal{B}}^h} ||\hat{u}^h - u^h||_{L^2}^2 \quad \text{s.t.} \quad \mathcal{I}[\hat{u}^h] = \mathcal{I}[u^h]. \quad (4.3.2)$$

Observe that

$$\begin{aligned} ||\hat{u}^h - u^h||_{L^2}^2 &= \sum_{i=0}^{\hat{M}} \sum_{j=0}^{\hat{M}} \hat{u}_i \hat{u}_j \hat{A}_{ij} - 2 \sum_{i=0}^{\hat{M}} \sum_{j=0}^{M} \hat{u}_i u_j^n C_{ij} + \sum_{i=0}^{M} \sum_{i=0}^{M} u_i^n u_j^n A_{ij} \\ &= \hat{\mathbf{u}}^T \hat{A} \hat{\mathbf{u}} - 2\hat{\mathbf{u}}^T C \mathbf{u}^n + \mathbf{u}^n A \mathbf{u}^n, \end{aligned}$$

where $A_{ij} = \langle \varphi_i, \varphi_j \rangle_{L^2}$, $\hat{A}_{ij} = \langle \hat{\varphi}_i, \hat{\varphi}_j \rangle_{L^2}$ and $C_{ij} = \langle \hat{\varphi}_i, \varphi_j \rangle_{L^2}$. The problem (4.3.2) can thus be reformulated as

$$\min_{\hat{\mathbf{u}} \in \mathbb{R}^{\hat{M}+1}} \hat{\mathbf{u}}^T \hat{A} \hat{\mathbf{u}} - 2\hat{\mathbf{u}}^T C \mathbf{u}^n + \mathbf{u}^n A \mathbf{u}^n \quad \text{s.t.} \quad \mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}) - \mathcal{I}_{\mathbf{p}^n}(\mathbf{u}^n) = 0.$$

This is a quadratic minimization problem with one nonlinear equality constraint, for which the solution $\hat{\mathbf{u}}$ is the solution of the nonlinear system of equations

$$\begin{aligned} \hat{A}\hat{\mathbf{u}} - C\mathbf{u}^n - \lambda \nabla \mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}) &= 0 \\ \mathcal{I}_{\mathbf{p}^{n+1}}(\hat{\mathbf{u}}) - \mathcal{I}_{\mathbf{p}^n}(\mathbf{u}^n) &= 0, \end{aligned}$$

which can be solved numerically using a suitable nonlinear solver.

4.4 Application to the BBM equation

4.4.1 The BBM equation

The BBM equation was introduced by Peregrine [19], and later studied by Benjamin et al. [2] as a model for small amplitude long waves on the surface of water in a channel. Conservative finite difference schemes for the BBM equation were proposed in [20] and [6], the latter being a discrete gradient method on fixed grids. A moving mesh FEM scheme employing a quasi-Lagrange approach is presented by Lu, Huang and Qiu in [16], which we also refer to for a more extensive list of references to the existing numerical schemes for the BBM equation.

Consider now an initial-boundary value problem of the one-dimensional BBM equation with periodic boundary conditions,

$$u_t - u_{xxt} + u_x + uu_x = 0, \qquad x \in [-L, L], \quad t \in (0, T] \tag{4.4.1}$$

$$u(x, 0) = u_0(x), \qquad x \in [-L, L] \tag{4.4.2}$$

$$u(-L, t) = u(L, t), \qquad t \in (0, T]. \tag{4.4.3}$$

By introducing the new variable $m(x, t) := u(x, t) - u_{xx}(x, t)$, equation (4.4.1) can be rewritten on the form (4.2.2) as

$$m_t = \mathcal{S}(m)\frac{\delta \mathcal{H}}{\delta m},$$

for two different pairs of an antisymmetric differential operator $\mathcal{S}(m)$ and a Hamiltonian $\mathcal{H}[m]$:

$$\begin{aligned} \mathcal{S}^1(m) &= -(\frac{2}{3}u + 1)\partial_x - \frac{1}{3}u_x, \\ \mathcal{H}^1[m] &= \frac{1}{2}\int (u^2 + u_x^2)\mathrm{d}x, \end{aligned}$$

and

$$\begin{aligned} \mathcal{S}^2(m) &= -\partial_x + \partial_{xxx}, \\ \mathcal{H}^2[m] &= \frac{1}{2}\int (u^2 + \frac{1}{3}u^3)\mathrm{d}x. \end{aligned}$$

4.4.2 Discrete schemes

We apply the PUM approach to create numerical schemes which preserve an approximation to either $\mathcal{H}^1[m]$ or $\mathcal{H}^2[m]$, splitting $\Omega := [-L, L]$ into M elements $\{[x_i, x_{i+1}]\}_{i=0}^{M-1}$. Defining the matrices A and E by their components

$$A_{ij} = \int_\Omega \varphi_i \varphi_j \mathrm{d}x \quad \text{and} \quad E_{ij} = \int_\Omega \varphi_{i,x} \varphi_{j,x} \mathrm{d}x,$$

we set $\mathbf{m} = (A + E)\mathbf{u}$. Note that the matrices A and E depend on the mesh, and thus will change when adaptivity is used. We will then distinguish between matrices from different time steps by writing e.g. A^n and A^{n+1}.

Approximating u by u^h as in section 4.2.2, we find

$$\begin{aligned}
\mathcal{H}^1_{\mathbf{p}}(\mathbf{m}) = \mathcal{H}^1[m^h] &= \frac{1}{2}\int_\Omega (u^h)^2 + (u^h_x)^2 \mathrm{d}x \\
&= \frac{1}{2}\sum_{i,j} u_i u_j \int_\Omega \varphi_i \varphi_j \mathrm{d}x + \frac{1}{2}\sum_{i,j} u_i u_j \int_\Omega \varphi_{i,x}\varphi_{j,x}\mathrm{d}x \\
&= \frac{1}{2}\mathbf{u}^{\mathrm{T}}(A + E)\mathbf{u}
\end{aligned}$$

The integrals can be evaluated exactly and efficiently by considering elementwise which basis functions are supported on the element before applying Gaussian quadrature to obtain exact evaluations of the polynomial integrals. We define the matrix $B_1(\mathbf{u})$ by

$$B_1(\mathbf{u})_{ji} = -\frac{2}{3}\sum_{k=0}^{M-1} u_k \left(2\int_\Omega \varphi_i \varphi_{j,x}\varphi_k \mathrm{d}x + \int_\Omega \varphi_i \varphi_j \varphi_{k,x}\mathrm{d}x\right) - \int_\Omega \varphi_i \varphi_{j,x}\mathrm{d}x.$$

An approximation to the gradient of $\mathcal{H}^1$ with respect to m is found by the AVF discrete gradient

$$\begin{aligned}
\overline{\nabla}\mathcal{H}^1_{\mathbf{p}}(\mathbf{m}^n, \mathbf{m}^{n+1}) &= (A + E)^{-1}\overline{\nabla}\mathcal{H}^1_{\mathbf{p}}(\mathbf{u}^n, \mathbf{u}^{n+1}) \\
&= (A + E)^{-1}\int_0^1 \nabla\mathcal{H}^1_{\mathbf{p}}(\xi\mathbf{u}^n + (1-\xi)\mathbf{u}^{n+1})\mathrm{d}\xi \\
&= (A + E)^{-1}\frac{1}{2}(A + E)\left(\mathbf{u}^n + \mathbf{u}^{n+1}\right) = \frac{1}{2}\left(\mathbf{u}^n + \mathbf{u}^{n+1}\right).
\end{aligned}$$

Thus we have the required terms for forming the system (4.2.4) and applying the adaptive discrete gradient method to it. Corresponding to (4.3.1), we get the scheme

$$\begin{aligned}
(A^{n+1} + E^{n+1})\left(\mathbf{u}^{n+1} - \hat{\mathbf{u}}\right) = &\frac{\hat{\mathbf{u}}^{\mathrm{T}}\left(A^{n+1} + E^{n+1}\right)\hat{\mathbf{u}} - \mathbf{u}^{n\mathrm{T}}\left(A^n + E^n\right)\mathbf{u}^n}{\left(\hat{\mathbf{u}} + \mathbf{u}^{n+1}\right)^{\mathrm{T}}\left(\hat{\mathbf{u}} + \mathbf{u}^{n+1}\right)}\left(\hat{\mathbf{u}} + \mathbf{u}^{n+1}\right) \\
&+ \frac{\Delta t}{2}B_1^{n+1}\left(\frac{\hat{\mathbf{u}} + \mathbf{u}^{n+1}}{2}\right)\left(\hat{\mathbf{u}} + \mathbf{u}^{n+1}\right),
\end{aligned}$$

where $\mathbf{v}^{n+1} = \hat{u} + \mathbf{u}^{n+1}$. Here we have chosen the skew-symmetric matrix B_1 to be a function of $\hat{\mathbf{u}}$ and $\mathbf{u}^{n+1}$, but could also have chosen e.g. $B_1(\hat{\mathbf{u}})$, resulting in a decreased computational cost at the expense of less precise results. During testing, the basis functions were chosen as piecewise cubic polynomials.

In the same manner we may obtain a scheme that preserves $\mathcal{H}^2[m]$. In this case

$$\begin{aligned}\mathcal{H}^2_{\mathbf{p}}(\mathbf{m}) = \mathcal{H}^2[m^h] &= \frac{1}{2}\int_\Omega (u^h)^2 + \frac{1}{3}(u^h)^3 \mathrm{d}x \\ &= \frac{1}{2}\sum_{i,j} u_i u_j \int_\Omega \varphi_i \varphi_j \mathrm{d}x + \frac{1}{6}\sum_{i,j,k} u_i u_j u_k \int_\Omega \varphi_i \varphi_j \varphi_k \mathrm{d}x.\end{aligned}$$

and

$$(B_2)_{ji} = -\int_\Omega \varphi_i \varphi_{j,x} \mathrm{d}x + \int_\Omega \varphi_i \varphi_{j,xxx} \mathrm{d}x.$$

Note that the skew-symmetric matrix B_2 is independent of $\mathbf{u}$.

Defining the tensor D by its elements

$$D_{ijk} = \int_\Omega \varphi_i \varphi_j \varphi_k \mathrm{d}x,$$

we get, with the convention of summation over repeated indices, the AVF discrete gradient with respect to $\mathbf{u}$ given by the elements

$$\overline{\nabla}\mathcal{H}^2_{\mathbf{p}}(\mathbf{u}^n, \mathbf{u}^{n+1})_i = \frac{A_{ij}}{2}(u_j^n + u_j^{n+1}) + \frac{D_{ijk}}{6}\left(u_j^n(u_k^n + \frac{u_k^{n+1}}{2}) + u_j^{n+1}(\frac{u_k^n}{2} + u_k^{n+1})\right)$$

and again the discrete gradient with respect to $\mathbf{m}$ by

$$\overline{\nabla}\mathcal{H}^2_{\mathbf{p}}(\mathbf{m}^n, \mathbf{m}^{n+1}) = (A+E)^{-1}\overline{\nabla}\mathcal{H}^2_{\mathbf{p}}(\mathbf{u}^n, \mathbf{u}^{n+1}).$$

If we employ integral preserving transfer between the meshes, we get the scheme

$$\mathbf{u}^{n+1} - \hat{\mathbf{u}} = \Delta t (A+E)^{-1} B_2 (A+E)^{-1} \overline{\nabla}\mathcal{H}^2_{\mathbf{p}}(\hat{\mathbf{u}}, \mathbf{u}^{n+1}),$$

where we note that $S_{\mathbf{p},2} := (A+E)^{-1}B_2(A+E)^{-1}$ is a skew-symmetric matrix. If non-preserving transfer is used, we need a correction term, as in the $\mathcal{H}^1$ scheme above. The calculation of such a term is straightforward, but we omit it here for reasons of brevity.

To approximate the third derivative in B_2, we need basis functions of at least degree three, and to guarantee skew-symmetry in B_2, these basis functions need to be C^2 on the element boundaries. This is not obtainable with regular nodal FEM basis functions, so we have instead used third order B-spline basis functions as described in [7] during testing.

4.5 Numerical Results

To demonstrate the performance of our methods, we have tested them on two one-dimensional simple problems: A soliton solution, and the interaction of two waves. We have tested our $\mathcal{H}^1$- and $\mathcal{H}^2$-preserving schemes on uniform and moving meshes, and compared the results to those obtained using the explicit midpoint method. For the transfer operation between meshes, we have used a piecewise cubic interpolation method in the $\mathcal{H}^1$ preserving scheme, and exact transfer in the $\mathcal{H}^2$ preserving scheme.

4.5.1 Mesh adaptivity

As noted in section 4.3, our methods can be coupled with any adaptive mesh strategy using the rezoning approach. For our numerical experiments, we have used a simple method for r-adaptivity based on the equidistribution principle: Splitting Ω into M intervals, we require that

$$\int_{x_i}^{x_{i+1}} \omega(x)\mathrm{d}x = \frac{1}{M}\int_{-L}^{L} \omega(x)\mathrm{d}x,$$

where the monitor function ω is a function measuring how densely grid points should lie, based on the value of u. For a general discussion on the choice of an optimal monitor function, see e.g. [3, 4]. For the problems we have studied, a generalized solution arc length monitor function proved to yield good results. This is given by

$$\omega(x) = \sqrt{1 + k^2\left(\frac{\partial u}{\partial x}(x)\right)^2}.$$

For $k = 1$, this is the usual arc length monitor function, in which case the equidistribution principle amounts to requiring that the arc length of u over each interval is equal. In applications, we only have an approximation of u, and hence ω must be approximated as well. We have applied a finite difference approximation and obtained approximately equidistributing grids using de Boor's method as explained in [14, pp. 36-38].

4.5.2 Soliton solution

With $u_0(x) = 3(c-1)\operatorname{sech}^2\left(\frac{1}{2}\sqrt{1-\frac{1}{c}}x\right)$, the exact solution of (4.4.1)–(4.4.3) is

$$u(x,t) = 3(c-1)\operatorname{sech}^2\left(\frac{1}{2}\sqrt{1-\frac{1}{c}}\,l(x,t)\right),$$

with $l(x,t) = \min_{j\in\mathbb{Z}} \left|x - ct + 2jL\right|$. This is a soliton solution which travels with a constant speed c in x-direction while maintaining its initial shape.

To evaluate the numerical solutions, we have compared them to the exact solution and calculated errors in shape and phase. The phase error is evaluated as

$$E_n^{\text{phase}} = |ct_n - x^*|,$$

where $x^* = \arg\max_x u_h(x, t_n)$, i.e. the location of the peak of the soliton in the numerical solution. The shape error is given by

$$E_n^{\text{shape}} = \left|\left| u_h(x, t_n) - u\left(x, \frac{x^*}{c}\right)\right|\right|,$$

where the peak of the exact solution is translated to match the peak of the numerical solution, and the difference in the shapes of the solitons is calculated.

The results of the numerical tests can be seen in figures 4.1–4.3. Here, M denotes the degrees of freedom used in the spatial approximation and Δt the fixed time step size. DG1 and DG1MM denotes the $\mathcal{H}_{\mathbf{p}}^1$ preserving scheme with fixed, uniform grid and adaptive grid, respectively; similary DG2 and DG2MM denotes the $\mathcal{H}_{\mathbf{p}}^2$ preserving scheme with uniform and adaptive grids.

In Figure 4.1 we see the relative errors in $\mathcal{H}_{\mathbf{p}}^1$ and $\mathcal{H}_{\mathbf{p}}^2$. The DG1 and DG1MM schemes are compared to schemes using the same 3rd order nodal basis functions, but the trapezoidal rule for time-stepping, denoted by TR and TRMM. Likewise, the DG2 and DG2MM schemes are compared to the IM and IMMM schemes, using B-spline basis functions and the implicit midpoint method for discretization in time. The error in $\mathcal{H}_{\mathbf{p}}^1$ is very small for the DG1 and DG1MM schemes, as expected. Also the error in $\mathcal{H}_{\mathbf{p}}^2$ is very small for the DG2 and DG2MM schemes. The order of the error is not machine precision, but is instead dictated by the precision with which the nonlinear equations in each time step is solved. We can also see that while the TR and IM schemes, with and without moving meshes, have poor conservation properties, the moving mesh DG schemes seem to preserve quite well even the integrals they are not designed to preserve.

In figures 4.2 and 4.3 we see the phase and shape errors, of our methods compared to non-moving mesh methods and non-preserving methods, respectively. The advantage of using moving meshes is clear, especially for the $\mathcal{H}_{\mathbf{p}}^2$ preserving schemes. The usefulness on integral preservation is ambiguous in this case. It seems that what we gain in precision in phase, we lose in precision in shape, and vice versa.

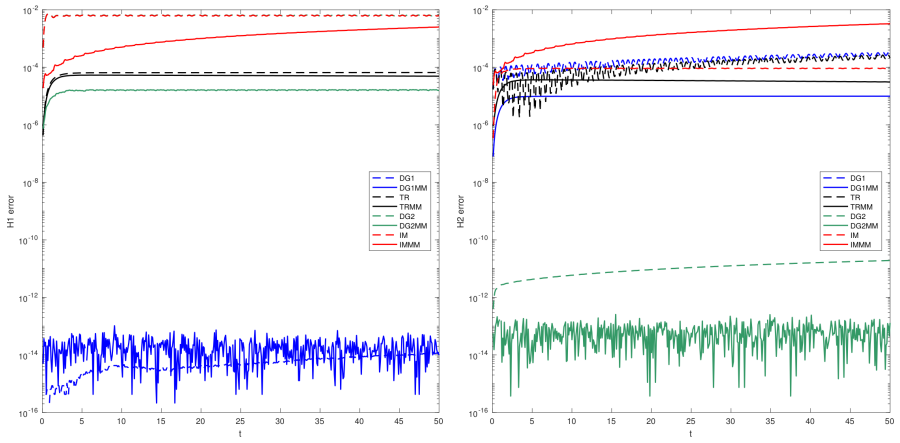


Figure 4.1: The soliton problem. Relative error in the approximated Hamiltonians $\mathcal{H}^1_{\mathbf{p}}$ (left) and $\mathcal{H}^2_{\mathbf{p}}$ (right) plotted as a function of time $t \in [0, 50]$. $c = 3, L = 200, \Delta t = 0.1$, $M = 200$.

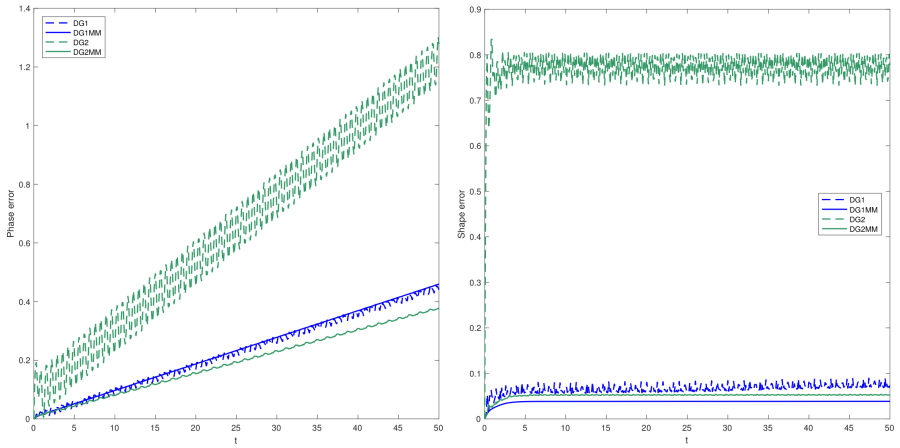


Figure 4.2: The soliton problem. Phase error (left) and shape error (right) as a function of time. $c = 3, L = 200, \Delta t = 0.1$, $M = 200$.

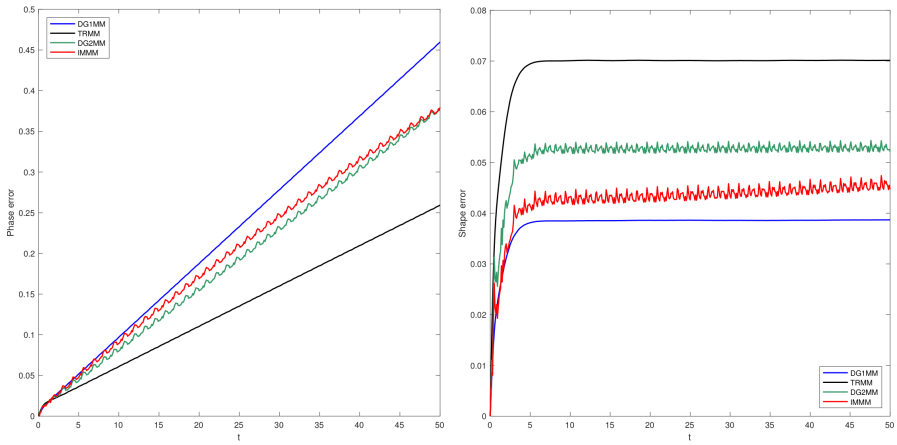


Figure 4.3: The soliton problem. Phase error (left) and shape error (right) as a function of time. $c = 3, L = 200, \Delta t = 0.1,\ M = 200$.

4.5.3 A small wave overtaken by a large one

A typical test problem for the BBM equation is the interaction between two solitary waves. With the initial condition

$$u_0(x) = 3(c_r - 1)\,\mathrm{sech}^2\left(\sqrt{1 - \frac{1}{c_r}}\,\frac{x - x_r}{2}\right) + 3(c_s - 1)\,\mathrm{sech}^2\left(\sqrt{1 - \frac{1}{c_s}}\,\frac{x - x_s}{2}\right),$$

one wave will eventually be overtaken by the other as long as $c_r \neq c_s$, i.e. if one wave is larger than the other. There is no available analytical solution for this problem. The two waves are not solitons, as the amplitudes will change a bit after the waves have interacted [9].

Solutions obtained by solving the problem with our two energy preserving schemes, giving very similar results, are plotted in Figure 4.4. Also, to illustrate the mesh adaptivity, we have included a plot of the mesh trajectories in Figure 4.6. Each line represents the trajectory of one mesh point in time, and we can see that the mesh points cluster nicely around the edges of the waves as they move.

To illustrate the performance of our methods, we have in Figure 4.5 compared solutions obtained by using the $\mathcal{H}^2_{\mathbf{p}}$-preserving moving mesh method with the solutions obtained by using a fourth order Runge–Kutta method on a static mesh, with the same, and quite few, degrees of freedom. The DG2MM solution is visibly closer to the solutions in Figure 4.4. The non-preserving RK scheme does a worse job of preserving the amplitude and speed of the waves compared to the DG2MM scheme, and we observe unwanted oscillations.

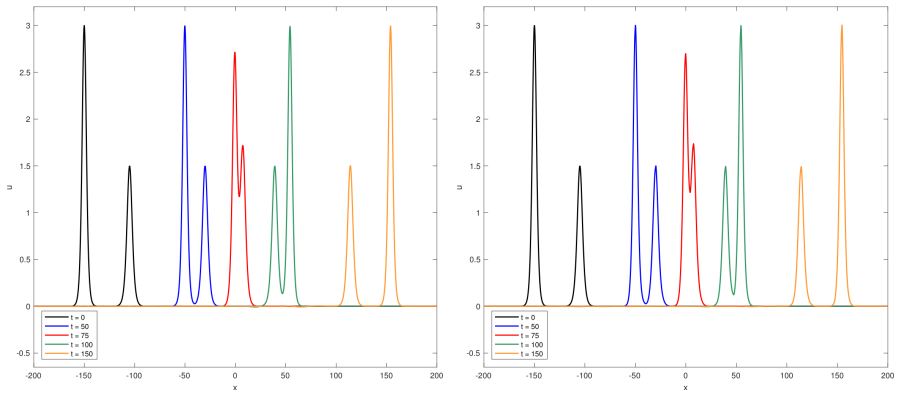


Figure 4.4: The interacting waves problem. Solutions at $t = \{0, 50, 75, 100, 150\}$ found by DG1MM (left) and DG2MM (right). $x_r = 150, x_s = 105, c_r = 2, c_s = 1.5, L = 200, \Delta t = 0.1$, $M = 1000$.

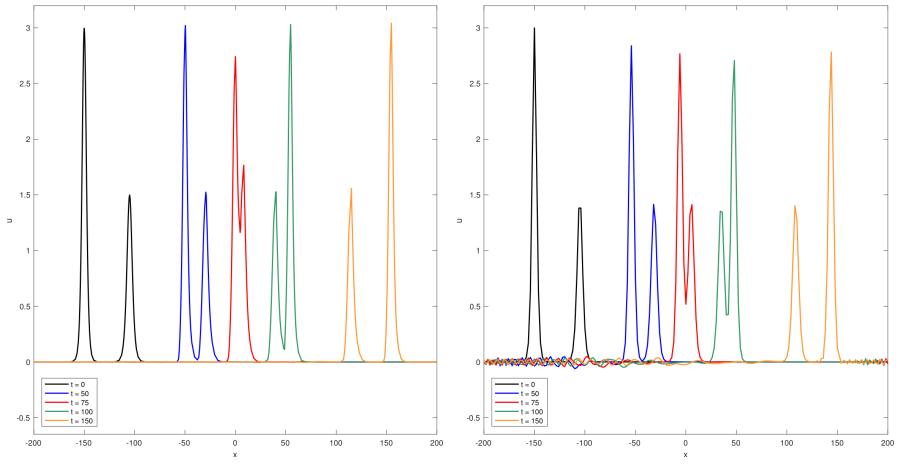


Figure 4.5: The interacting waves problem. Solutions at $t = \{0, 50, 75, 100, 150\}$ found by DG2MM (left) and RK (right). $x_r = 150, x_s = 105, c_r = 2, c_s = 1.5, L = 200, \Delta t = 0.1$, $M = 200$.

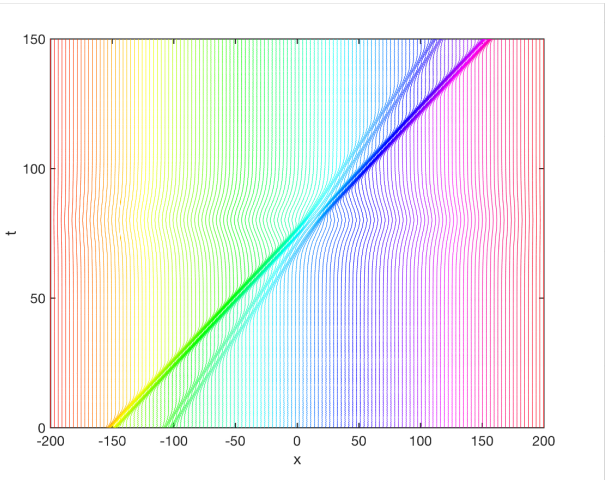


Figure 4.6: Mesh point trajectories in time. Each line represents one mesh point.

In Figure 4.7 we have plotted the Hamiltonian errors for this problem. Again we see that the energy preserving schemes preserve both Hamiltonians better than the Runge–Kutta scheme, but we do also observe that the DG1 scheme preserves $\mathcal{H}^2_{\mathbf{p}}$ better than the DG1MM scheme, and vice versa for the DG2 and DG2MM schemes. Note also that an increase in the errors can be observed when the two waves interact, but that this increase is temporary.

4.6 Conclusion

In this paper, we have presented energy preserving schemes for a class of PDEs, first on general fixed meshes, and then on adaptive meshes. These schemes are then applied to the BBM equation, for which discrete schemes preserving two of the Hamiltonians of the problem are explicitly given.

Numerical experiments are performed, using the energy preserving moving mesh schemes on two different BBM problems: a soliton solution, and two waves interacting. Plots of the phase and shape errors illustrate how, for the given parameters, the usage of moving meshes gives improved accuracy, while the integral preservation gives comparable results to existing methods, without yielding a categorical improvement. We will remark, however, that in many cases, the preservation of a quantity such as one of the Hamiltonians in itself may be a desired property of a numerical scheme. For the two wave interaction problem, we do not have an analytical solution to compare to, but plots of the solution indicate that our schemes perform well compared to a Runge–Kutta scheme.

Although only one-dimensional problems are presented as numerical ex-

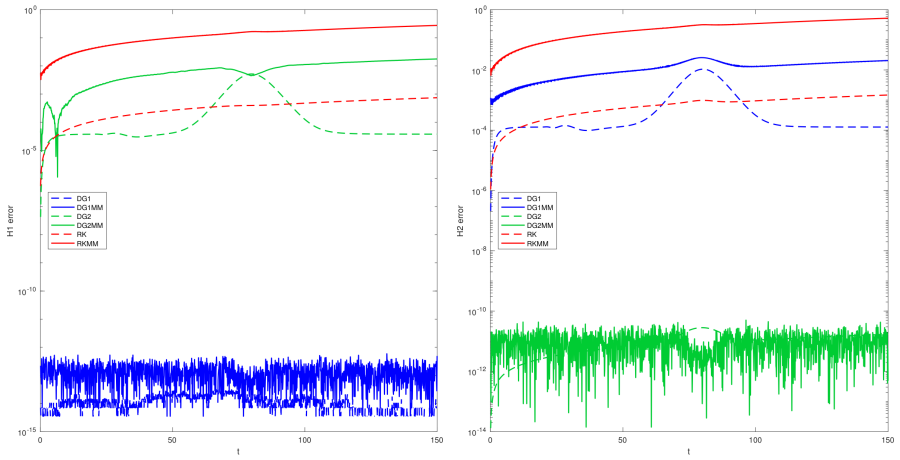


Figure 4.7: The interacting waves problem. Error in the approximated Hamiltonians $\mathcal{H}^1_{\mathbf{p}}$ (left) and $\mathcal{H}^2_{\mathbf{p}}$ (right) plotted as a function of time $t \in [0, 150]$. $x_r = 150, x_s = 105, c_r = 2, c_s = 1.5, L = 200, \Delta t = 0.1$, $M = 1000$.

amples here, the adaptive discrete gradient methods can be applied to multi-dimensional problems. This could be an interesting direction for further work, since the advantages of adaptive meshes are typically more evident when increasing the number of dimensions.

Bibliography

[1] I. Babuška and B. Guo, *The h, p and h-p version of the finite element method; basis theory and applications*, Adv. Eng. Softw., 15 (1992), pp. 159–174.

[2] T. B. Benjamin, J. L. Bona, and J. J. Mahony, *Model equations for long waves in nonlinear dispersive systems*, Philos. T. R. Soc. A., 272 (1972), pp. 47–78.

[3] J. Blom and J. Verwer, *On the use of the arclength and curvature monitor in a moving-grid method which is based on the method of lines*, tech. rep., NM-N8902, CWI, Amsterdam, 1989.

[4] C. J. Budd, W. Huang, and R. D. Russell, *Adaptivity with moving grids*, Acta Numer., 18 (2009), pp. 111–241.

[5] E. Celledoni, V. Grimm, R. I. McLachlan, D. I. McLaren, D. O'Neale, B. Owren, and G. R. W. Quispel, *Preserving energy*

resp. dissipation in numerical PDEs using the "average vector field" method, J. Comput. Phys., 231 (2012), pp. 6770–6789.

[6] D. COHEN AND X. RAYNAUD, *Geometric finite difference schemes for the generalized hyperelastic-rod wave equation*, J. Comput. Appl. Math., 235 (2011), pp. 1925–1940.

[7] J. A. COTTRELL, T. J. HUGHES, AND Y. BAZILEVS, *Isogeometric Analysis*, Wiley, 2009.

[8] R. COURANT, K. FRIEDRICHS, AND H. LEWY, *Über die partiellen Differenzengleichungen der mathematischen Physik*, Math. Ann., 100 (1928), pp. 32–74.

[9] W. CRAIG, P. GUYENNE, J. HAMMACK, D. HENDERSON, AND C. SULEM, *Solitary water wave interactions*, Phys. Fluids, 18 (2006), p. 057106.

[10] S. EIDNES, B. OWREN, AND T. RINGHOLM, *Adaptive energy preserving methods for partial differential equations*, Adv. Comput. Math., (2017).

[11] D. FURIHATA AND T. MATSUO, *Discrete variational derivative method*, Chapman & Hall/CRC Numerical Analysis and Scientific Computing, CRC Press, Boca Raton, FL, 2011.

[12] O. GONZALEZ, *Time integration and discrete Hamiltonian systems*, J. Nonlinear Sci., 6 (1996), pp. 449–467.

[13] E. HAIRER, C. LUBICH, AND G. WANNER, *Geometric numerical integration*, vol. 31 of Springer Series in Computational Mathematics, Springer, Heidelberg, 2010.

[14] W. HUANG AND R. RUSSELL, *Adaptive moving mesh methods*, vol. 174 of Springer Series in Applied Mathematical Sciences, Springer-Verlag, New York, 2010.

[15] S. LI AND L. VU-QUOC, *Finite difference calculus invariant structure of a class of algorithms for the nonlinear Klein-Gordon equation*, SIAM J. Numer. Anal., 32 (1995), pp. 1839–1875.

[16] C. LU, W. HUANG, AND J. QIU, *An adaptive moving mesh finite element solution of the regularized long wave equation*, J. Sci. Comput., 74 (2018), pp. 122–144.

[17] R. I. MCLACHLAN, G. R. W. QUISPEL, AND N. ROBIDOUX, *Geometric integration using discrete gradients*, Philos. T. R. Soc. A, 357 (1999), pp. 1021–1045.

[18] Y. MIYATAKE AND T. MATSUO, *A note on the adaptive conservative/dissipative discretization for evolutionary partial differential equations*, J. Comput. Appl. Math., 274 (2015), pp. 79–87.

[19] D. PEREGRINE, *Calculations of the development of an undular bore*, J. Fluid Mech., 25 (1966), pp. 321–330.

[20] T.-C. WANG AND L.-M. ZHANG, *New conservative schemes for regularized long wave equation*, Numer. Math. Chin., 15 (2006), pp. 348–356.

[21] T. YAGUCHI, T. MATSUO, AND M. SUGIHARA, *An extension of the discrete variational method to nonuniform grids*, J. Comput. Phys., 229 (2010), pp. 4382–4423.

[22] T. YAGUCHI, T. MATSUO, AND M. SUGIHARA, *The discrete variational derivative method based on discrete differential forms*, J. Comput. Phys., 231 (2012), pp. 3963–3986.

Variational image analysis with Euler's elastica using a discrete gradient scheme

Torbjørn Ringholm, Jasmina Lazić and Carola-Bibiane Schönlieb

Submitted

Variational image analysis with Euler's elastica using a discrete gradient scheme

Abstract. This paper concerns an optimization algorithm for unconstrained non-convex problems where the objective function has sparse connections between the unknowns. The algorithm is based on applying a dissipation preserving numerical integrator, the Itoh–Abe discrete gradient scheme, to the gradient flow of an objective function, guaranteeing energy decrease regardless of step size. We introduce the algorithm, prove a convergence rate estimate for non-convex problems with Lipschitz continuous gradients, and show that convergence rates are independent of problem size if the objective function has sparse connections between unknowns. The algorithm is presented in serial and parallel versions. Numerical tests show its use in Euler's elastica regularized imaging problems and its convergence rate, and compares the execution time of the method to that of the iPiano algorithm and the gradient descent and Heavy-ball algorithms.

5.1 Introduction

A classic idea for minimizing a differentiable $V : \mathbb{R}^n \to \mathbb{R}$, $n \geq 1$, is considering its gradient flow

$$\dot{\mathbf{u}}(t) = -\nabla V(\mathbf{u}(t)) \tag{5.1.1}$$

and numerically integrating a solution along it. For example, the gradient descent algorithm can be easily derived from the explicit Euler scheme

$$\mathbf{u}^{k+1} - \mathbf{u}^k = -\tau \nabla V(\mathbf{u}^k),$$

where τ is a step size and $\mathbf{u}^k$ an approximation to the value of $\mathbf{u}(k\tau)$. Other schemes can be used, such as Runge-Kutta and multistep methods. A discussion on step size conditions under which algebraically stable Runge-Kutta methods are dissipative can be found in [13]. Even though these classes of ODE integrators are readily available, their use does not appear to have gained much traction in the optimization community.

This disregard may be attributed to a division between the goals of numerical integration and numerical optimization; whereas the ODE integration schemes seek to approximate a solution path of (5.1.1) as accurately as possible, optimization schemes try to find a stationary point of (5.1.1) as quickly as possible. The former task requires small time steps while the latter task is generally completed more efficiently the larger the time steps are. Thus, using regular ODE schemes to solve (5.1.1) is, in general, ineffective. However, in

a recent article [27], the authors demonstrate that several well-known efficient optimization methods can be deduced from ODE integration schemes applied to equation (5.1.1). Examples include Polyak's Heavy-ball method [24], which may also be interpreted as a discretization of a gradient flow with an inertial term, and Nesterov's accelerated gradient method [18]. These methods are equivalent to linear two-step methods with certain choices of step length. Also, the proximal point and proximal gradient methods [20] are shown to correspond to an implicit Euler method and an Implicit-Explicit scheme, respectively. This gives credibility to the idea that ODE solvers with certain properties may indeed be useful as optimization schemes.

In recent years, new ODE solvers with properties well suited to optimization have emerged. In [12], building on developments in the field of geometric integration, the authors apply discrete gradient schemes to the gradient flow of energy functionals arising from problems in variational image analysis. Discrete gradients, introduced in [11] and further studied in [17] have a property which is interesting from an optimization viewpoint; they are *dissipativity preserving*. When applied to a dissipative ODE such as a gradient flow, the resulting time steps are also dissipative in the sense that

$$V(\mathbf{u}^{k+1}) \leq V(\mathbf{u}^k).$$

The schemes thus convergence monotonously toward a critical point V^* regardless of the step size used in the numerical integration if V is continuously differentiable [12].

While very efficient solvers exist for convex optimization problems, see e.g. [8, 21, 29], the picture is different for non-convex optimization problems. Considerable effort has been spent in developing efficient schemes for classes of problems with special structure, e.g. problems with one convex but non-differentiable term and one non-convex but differentiable term [4, 23]. We will add to this effort by introducing a method based on the Itoh–Abe discrete gradient method [15] that is most effective when the objective function is continuously differentiable with sparsely connected unknowns. Indeed, in [12], the authors use discrete gradient schemes with non-convex problems in mind, so this paper may be viewed as a continuation of their work.

A problem that fits the format of being non-convex with sparse connections is variational image analysis using a discretized Euler's elastica functional as a regularizer. Introduced in [22] for de-occluding objects in images, Euler's elastica regularization was further analyzed and applied to inpainting problems by Chan, Kang, and Shen in [28], who derive the Euler-Lagrange equations for the continuous Euler's elastica functional and solve these via finite difference schemes. This approach is not very computationally efficient, and attempts have been made to create more effective schemes, in particular in [30] where

an augmented Lagrangian approach was considered. This approach was later refined in [36], [34], [1] and [35]. Also of note are the approaches in [5] and [9] where convex approximations to the objective function are considered.

The method presented in Algorithm 5.1 resembles the coordinate gradient descent method since it uses coordinatewise updates, except that the gradient is approximated by a discrete gradient. It is derivative free and easy to use with only a step size parameter to choose. The discrete gradient approximation guarantees decrease at each iteration, at the expense of using an implicit scheme. A recent survey of coordinate descent algorithms and their convergence can be found in [32]. According to this survey, for coordinate descent methods one can expect $V(x^k) - V^* \in \mathcal{O}(1/k)$ for convex problems, with linear convergence if the problem is strongly convex. In [19], an accelerated coordinate descent method is presented, with $V(x^k) - V^* \in \mathcal{O}(1/k^2)$ for convex problems at the expense of computing a full vector operation for each coordinate update. In the following, we will prove a convergence rate of $\min_{1\leq j\leq k}\left\{\|\nabla V(\mathbf{u}^j)\|^2\right\} \in \mathcal{O}(1/k^{1/2})$ for nonconvex, Lipschitz continuously differentiable problems. In [10], an $\mathcal{O}(1/k)$ convergence rate for convex, smooth problems and linear convergence for problems satisfying the Polyak-Łojasiewicz condition [16] are proved for several discrete gradient algorithm, including the one considered here. In the case of the Itoh–Abe discrete gradient, the rates have an $\mathcal{O}(n^{1/2})$ dependence on the problem size n; in Lemma 5.1 we show that the rates are, in fact, independent of n in the case when V has sparsely connected unknowns. We also propose a method for adaptive time stepping that allows a certain acceleration of the algorithm for differentiable V, using four additional parameters.

The paper is organized as follows: In the following section, we introduce discrete gradient methods for optimization and discuss the convergence rate and acceleration of a specific discrete gradient-type scheme. In section 3, the Euler's elastica regularization problem is introduced, and parallelization of the discrete gradient algorithm is discussed together with the effect of sparsity in the problem. Section 4 contains numerical experiments concerning the quality of denoising and inpainting, experimental convergence rates, the effect of coordinate ordering and problem size, execution time, and dependence on the initial condition. The final section summarizes the results.

5.2 Discrete gradient methods

The task at hand is to minimize a $\mathcal{C}^1$ functional $V : \mathbb{R}^n \to \mathbb{R}$, also called an energy, by solving its gradient flow

$$\dot{\mathbf{u}} = -\nabla V(\mathbf{u}), \tag{5.2.1}$$

where $\mathbf{u}(t) \in \mathbb{R}^n$ is the unknown and $\dot{\mathbf{u}}(t)$ denotes its time derivative. The reason for this is that V dissipates along the flow of (5.2.1); if $\mathbf{u}(t)$ solves (5.2.1), then

$$\frac{\mathrm{d}}{\mathrm{d}t} V(\mathbf{u}(t)) = \left\langle \dot{\mathbf{u}}, \nabla V(\mathbf{u}(t)) \right\rangle = -\|\nabla V(\mathbf{u}(t))\|^2 \leq 0,$$

where $\|\cdot\|$ denotes the Euclidian norm on $\mathbb{R}^n$. Due to the dissipation, $\mathbf{u}(t)$ approaches a critical point of V as $t \to \infty$ as long as V is bounded from below. In general, V is nonlinear such that numerical schemes must be employed to solve (5.2.1) until a large stopping time T. This gives rise to different optimization algorithms depending on the scheme used. For example, a forward Euler scheme results in the gradient descent method. The forward Euler method has several drawbacks, one being that choosing too large step sizes results in instability. This necessitates step size selection, which may result in impractically small steps considering that we wish to obtain a stationary point of (5.2.1). It is therefore of interest to investigate the use of numerical schemes that have lenient step size restrictions or none at all. One such class of schemes is called *discrete gradient* methods. Discrete gradients were introduced in [11] to unite several energy preserving and dissipative ODE solvers under a single label. A seminal paper [17] covers their use as ODE solvers and which ODEs they are applicable to.

Definition 5.1. Given a differentiable function $V : \mathbb{R}^n \to \mathbb{R}$, we say that $\overline{\nabla} V : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$ is a discrete gradient of V if it is continuous and for all $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$,

$$\left\langle \overline{\nabla} V(\mathbf{u}, \mathbf{v}), \mathbf{v} - \mathbf{u} \right\rangle = V(\mathbf{v}) - V(\mathbf{u}),$$
$$\lim_{\mathbf{v} \to \mathbf{u}} \overline{\nabla} V(\mathbf{u}, \mathbf{v}) = \nabla V(\mathbf{u}).$$

Discrete gradients can be used in schemes to solve (5.2.1) numerically by computing

$$\mathbf{u}^{k+1} = \mathbf{u}^k - \tau_k \overline{\nabla} V(\mathbf{u}^k, \mathbf{u}^{k+1}), \tag{5.2.2}$$

where $\tau_k > 0$ is the step size at iteration number k. A key property is that the scheme is dissipating; by Definition 5.1 and the scheme (5.2.2), we have

$$V(\mathbf{u}^{k+1}) - V(\mathbf{u}^k) = \left\langle \overline{\nabla} V(\mathbf{u}^k, \mathbf{u}^{k+1}), \mathbf{u}^{k+1} - \mathbf{u}^k \right\rangle = -\frac{1}{\tau_k} \|\mathbf{u}^{k+1} - \mathbf{u}^k\|^2. \tag{5.2.3}$$

Note that the dissipation property holds regardless of the step size τ_k.

Definition 5.1 is quite broad and as a result, there exist several types of discrete gradients. Two popular choices are the midpoint discrete gradient [11] and the Average Vector Field (AVF) discrete gradient [14]. They give second-order accurate schemes for (5.2.1) and are suited for solving ODEs precisely.

In our case, solving (5.2.1) as exactly as possible is not the main concern, rather, we need a scheme with cheap time steps and fast convergence toward a minimizer. The schemes obtained using the Gonzalez and AVF discrete gradients are fully implicit in the sense that in general, at each time step of (5.2.2), a single n-dimensional system of nonlinear equations must be solved. For large n, this is slow since the complexity of solving such a system is typically $\mathcal{O}(n^2)$. Instead, we consider the Itoh–Abe discrete gradient [15], defined componentwise as

$$(\overline{\nabla} V(\mathbf{u},\mathbf{v}))_l = \frac{V\left(\mathbf{u} + \sum_{j=1}^{l} (v_j - u_j)\mathbf{e}_j\right) - V\left(\mathbf{u} + \sum_{j=1}^{l-1} (v_j - u_j)\mathbf{e}_j\right)}{v_l - u_l},$$

where $\mathbf{e_j}$ denotes the j'th standard basis vector. This discrete gradient, while still implicit, has two advantages over the Gonzales and AVF discrete gradients. First, its use in the scheme (5.2.2) requires the solution of n scalar nonlinear equations per time step, meaning its computational complexity scales as $\mathcal{O}(n)$. Secondly, it is derivative-free and requires only computations of differences between the objective function with variation in one variable, which may be considerably cheaper than evaluating the objective function itself; consider for example the optimization problem

$$\min_{\mathbf{u}\in\mathbb{R}^n} \left\{ f(\mathbf{u}) = \sum_{i=1}^{M} f_i(\mathbf{u}) \right\},$$

where at most N of the f_i depend on a given coordinate, say, u_k. Then, computing the difference $f(\mathbf{u} + \mathbf{e}_k u_k) - f(\mathbf{u})$ amounts to computing at most N values of the f_i, a cost comparable to that of calculating one coordinate derivative of f.

5.2.1 The algorithm

The algorithm based on using the Itoh–Abe discrete gradient with fixed step size $\tau_k = \tau$ in (5.2.2) is presented in Algorithm 5.1. As a stopping criterion we set a tolerance tol and stop when $(V(\mathbf{u}^k) - V(\mathbf{u}^{k-1}))/V(\mathbf{u}^0) < tol$. This criterion is economical to evaluate, requiring no evaluation of V since the energy increments are known from the dissipation property (5.2.3).

Algorithm 5.1. DG

Choose $\tau > 0$, $tol > 0$ and $\mathbf{u}^0 \in \mathbb{R}^n$. Set $k = 0$.
repeat
  $\mathbf{v}_0^k = \mathbf{u}^k$
  for $j = 1, ..., n$ **do**

Solve $\beta_j^k = -\tau(V(\mathbf{v}_{j-1}^k + \beta_j^k \mathbf{e}_j) - V(\mathbf{v}_{j-1}^k))/\beta_j^k$
$\mathbf{v}_j^k = \mathbf{v}_{j-1}^k + \beta_j^k \mathbf{e}_j$
end for
$\mathbf{u}^{k+1} = \mathbf{v}_n^k$
$k = k+1$
until $\left(V(\mathbf{u}^k) - V(\mathbf{u}^{k-1})\right)/V(\mathbf{u}^0) < tol$

To solve the nonlinear scalar subproblems defining the β_j^k, we use the Brent-Dekker algorithm [6]. This is a derivative free method based on a combination of bisection and interpolation algorithms which converges superlinearly if the function whose root is to be found is $\mathcal{C}^1$ near the root. It is the method of choice for scalar root finding problems in [25].

The algorithm can be accelerated through adaptive step sizes. Unlike line search methods, each time step is implicit and so changing τ_k requires a recomputation of $\mathbf{u}^{k+1}$, which can be costly and should be avoided. One way of adapting τ_k, which can be used for differentiable V, is to check conditions similar to the Wolfe conditions [31]. We consider, with constants $c_1 \in (0,1)$ and $c_2 \in (c_1, 1)$ the conditions

$$V(\mathbf{u}^{k+1}) - V(\mathbf{u}^k) \leq c_1 \left\langle \nabla V(\mathbf{u}^k), \mathbf{u}^{k+1} - \mathbf{u}^k \right\rangle, \tag{5.2.4}$$

$$\left\langle \nabla V(\mathbf{u}^{k+1}), \mathbf{u}^{k+1} - \mathbf{u}^k \right\rangle \geq c_2 \left\langle \nabla V(\mathbf{u}^k), \mathbf{u}^{k+1} - \mathbf{u}^k \right\rangle. \tag{5.2.5}$$

If condition (5.2.4) holds, regardless of whether (5.2.5) holds, then τ_k is increased for the next iteration by a factor $\lambda > 1$. If (5.2.4) does not hold but (5.2.5) does, one takes $\tau_{k+1} = \rho\tau_k$ where $\rho \in (0,1)$. If neither condition holds, the step size is not changed. In all cases, the new value $\mathbf{u}^{k+1}$ is accepted. With this approach one obtains step sizes that are adjusted based on prior performance while not wasting previous computations, summed up in Algorithm 5.2.

Algorithm 5.2. DG-ADAPT
Choose $\tau_0 > 0$, $tol > 0$, $\rho \in (0,1)$, $\lambda > 1$, $c_1 \in (0,1)$, $c_2 \in (c_1, 1)$, and $\mathbf{u}^0 \in \mathbb{R}^n$.
Set $k = 0$.
repeat
$\mathbf{v}_0^k = \mathbf{u}^k$
for $j = 1, ..., n$ **do**
Solve $\beta_j^k = -\tau_k(V(\mathbf{v}_{j-1}^k + \beta_j^k \mathbf{e}_j) - V(\mathbf{v}_{j-1}^k))/\beta_j^k$
$\mathbf{v}_j^k = \mathbf{v}_{j-1}^k + \beta_j^k \mathbf{e}_j$
end for
$\mathbf{u}^{k+1} = \mathbf{v}_n^k$
if $V(\mathbf{u}^{k+1}) - V(\mathbf{u}^k) \leq c_1 \left\langle \nabla V(\mathbf{u}^k), \mathbf{u}^{k+1} - \mathbf{u}^k \right\rangle$ **then**

$\tau_{k+1} = \lambda\tau_k$
else if $\left\langle \nabla V(\mathbf{u}^{k+1}), \mathbf{u}^{k+1} - \mathbf{u}^k \right\rangle \geq c_2 \left\langle \nabla V(\mathbf{u}^k), \mathbf{u}^{k+1} - \mathbf{u}^k \right\rangle$ **then**
$\tau_{k+1} = \rho\tau_k$
end if
$k = k + 1$
until $\left(V(\mathbf{u}^k) - V(\mathbf{u}^{k-1})\right) / V(\mathbf{u}^0) < tol$

Condition (5.2.5) provides a lower bound on the step size when ∇V is Lipschitz continuous. Firstly, since ∇V is Lipschitz, the descent lemma [3, Proposition A.24], provides the estimate

$$V(\mathbf{u}) \leq V(\mathbf{v}) + \left\langle \nabla V(\mathbf{v}), \mathbf{u} - \mathbf{v} \right\rangle + \frac{L}{2}\|\mathbf{v} - \mathbf{u}\|^2 \tag{5.2.6}$$

which holds for all $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$. Combining (5.2.6) with (5.2.5) and (5.2.3) we find

$$\begin{aligned} c_2 \left\langle \nabla V(\mathbf{u}^k), \mathbf{u}^k - \mathbf{u}^{k+1} \right\rangle &\geq \left\langle \nabla V(\mathbf{u}^{k+1}), \mathbf{u}^k - \mathbf{u}^{k+1} \right\rangle \\ &\geq V(\mathbf{u}^k) - V(\mathbf{u}^{k+1}) - \frac{L}{2}\|\mathbf{u}^{k+1} - \mathbf{u}^k\|^2 \\ &= \left(1 - \frac{L}{2}\tau_k\right)\left(V(\mathbf{u}^k) - V(\mathbf{u}^{k+1})\right). \end{aligned}$$

Rearranging and applying (5.2.6) once more, we find

$$\frac{L}{2}\tau_k \left(V(\mathbf{u}^k) - V(\mathbf{u}^{k+1})\right) \geq (1 - c_2)\left(V(\mathbf{u}^k) - V(\mathbf{u}^{k+1})\right) - \frac{c_2 L}{2}\|\mathbf{u}^{k+1} - \mathbf{u}^k\|^2.$$

Using (5.2.3) on the last term to eliminate $V(\mathbf{u}^k) - V(\mathbf{u}^{k+1})$ we find the lower bound

$$\tau_k \geq \frac{1 - c_2}{1 + c_2}\frac{2}{L}.$$

5.2.2 Convergence

In [12], the authors prove that the iterates of Algorithm 5.1 converge toward a critical point, but do not estimate the convergence rate. Theorem 5.1 concerns the convergence rate of Algorithm 5.2 in the general case of a non-convex objective V. A sublinear convergence rate of $\mathcal{O}(1/k)$ for convex problems and a linear convergence rate for problems satisfying the Polyak-Łojasiewicz inequality are given in [10]. These theorems are stated below without proof to support the discussion of numerical results presented in section 5.4, where better rates than the ones proved in Theorem 5.1 are observed when choosing ϵ in the smoothing of the Euler's elastica regularizer large enough. The following assumption is common to these theorems and Lemma 5.1 in the subsequent section.

Assumption 5.1. *The function $V : \mathbb{R}^n \to \mathbb{R}$ is $\mathcal{C}^1$, bounded from below and coercive. Furthermore, ∇V is Lipschitz with Lipschitz constant L and coordinatewise Lipschitz constants $L_i \in [L_{\min}, L_{\max}]$, and all time steps τ_j lie in $[\tau_{\min}, \tau_{\max}] \subset \mathbb{R}^+$.*

The following proof is inspired by that of [2, Lemma 3.3].

Theorem 5.1. *If Assumption 5.1 holds, the $\mathbf{u}^k$ produced by Algorithm 5.2 satisfy*

$$\min_{1\le j\le k}\left\{\|\nabla V(\mathbf{u}^j)\|^2\right\} \le \nu\frac{V(\mathbf{u}^0)-V^*}{k}, \quad \nu = 2L_{\max}^2\left(\tau_{\max}n + \frac{\tau_{\max}}{L_{\max}^2\tau_{\min}^2}\right)$$

where $V^* > -\infty$ is a local minimum.

Proof. The coordinatewise Itoh–Abe scheme, with $\beta_l^k = u_l^{k+1} - u_l^k$, reads

$$\beta_l^k = -\tau_k\frac{V\left(\mathbf{v}_l^k\right) - V\left(\mathbf{v}_{l-1}^k\right)}{\beta_l^k},$$

with $\mathbf{v}_l^k := \mathbf{u}^k + \sum_{j=1}^{l}\beta_j^k\mathbf{e}_j$ such that $\mathbf{u}^{k+1} = \mathbf{v}_n^k$. By the triangle inequality,

$$\left|\frac{\partial V}{\partial u_l}(\mathbf{u}^{k+1})\right| \le \left|\frac{\partial V}{\partial u_l}(\mathbf{u}^{k+1}) - \frac{V(\mathbf{v}_l^k) - V(\mathbf{v}_{l-1}^k)}{\beta_l^k}\right| + \frac{1}{\tau_k}\left|\beta_l^k\right|.$$

Since $V \in \mathcal{C}^1$, the mean value theorem holds, meaning

$$\frac{V(\mathbf{v}_l^k) - V(\mathbf{v}_{l-1}^k)}{\beta_l^k} = \frac{\partial V}{\partial u_l}(\mathbf{v}_{l-1}^k + s\beta_l^k\mathbf{e}_l)$$

for some $s \in (0,1)$. Hence,

$$\left|\frac{\partial V}{\partial u_l}(\mathbf{u}^{k+1})\right| \le \left|\frac{\partial V}{\partial u_l}(\mathbf{u}^{k+1}) - \frac{\partial V}{\partial u_l}(\mathbf{v}_{l-1}^k + s\beta_l^k\mathbf{e}_l)\right| + \frac{1}{\tau_k}\left|\beta_l^k\right|. \tag{5.2.7}$$

Exploiting the coordinatewise Lipschitz continuity of the gradient, we have

$$\left|\frac{\partial V}{\partial u_l}(\mathbf{u}^{k+1})\right| \le L_l\|\mathbf{u}^{k+1} - \mathbf{v}_{l-1}^k - s\beta_l^k\mathbf{e}_l\| + \frac{1}{\tau_k}\left|\beta_l^k\right|.$$

Squaring this and summing over all coordinates, we get

$$\begin{aligned}
\|\nabla V(\mathbf{u}^{k+1})\|^2 &\le \sum_{l=1}^{n}\left(L_l\|\mathbf{u}^{k+1} - \mathbf{v}_{l-1}^k - s\beta_l^k\mathbf{e}_l\| + \frac{1}{\tau_k}\left|\beta_l^k\right|\right)^2 \\
&\le 2L_{\max}^2\left(n + \frac{1}{L_{\max}^2\tau_{\min}^2}\right)\|\mathbf{u}^{k+1} - \mathbf{u}^k\|^2 \\
&\le \nu\left(V(\mathbf{u}^k) - V(\mathbf{u}^{k+1})\right).
\end{aligned}$$

We then find

$$k \min_{1\le j\le k} \left\{ \|\nabla V(\mathbf{u}^j)\|^2 \right\} \le \sum_{j=1}^{k} \|\nabla V(\mathbf{u}^j)\|^2 \le \nu \left(V(\mathbf{u}^0) - V(\mathbf{u}^k) \right) \le \nu \left(V(\mathbf{u}^0) - V^* \right),$$

which concludes the proof. □

Remark: We can choose a fixed $\tau_k = \tau$ that minimizes ν, yielding

$$\tau = \frac{1}{L_{\max}\sqrt{n}}, \quad \nu = 4L_{\max}\sqrt{n}.$$

Thus, the complexity of the above bound with respect to the problem size n is $\mathcal{O}(n^{1/2})$. Furthermore, we can obtain bounds of the type $\|\nabla V(\mathbf{u}^{k+1})\|^2 \le \nu \left(V(\mathbf{u}^k) - V(\mathbf{u}^{k+1}) \right)$ for the Gonzalez and AVF discrete gradients, yielding similar convergence rates for these discrete gradients. Such estimates are shown in [10, Lemma 5.1].

As with descent methods, the convergence rate improves with additional assumptions on V, in particular assuming that V is convex. We state the following theorems, proved in [10] and inspired by those in [2], for later reference. Similarly to the proof of Theorem 5.1, they are are based on bounding $\|\nabla V(\mathbf{u}^{k+1})\|^2 \le \nu \left(V(\mathbf{u}^k) - V(\mathbf{u}^{k+1}) \right)$ and so the factor ν appears here as well.

Theorem 5.2. *If Assumption 5.1 holds and V is in addition convex, the iterates $\mathbf{u}^k$ produced by Algorithm 5.2 satisfy, with ν as in Theorem 5.1,*

$$V(\mathbf{u}^k) - V^* \le \frac{\nu R(\mathbf{u}^0)^2}{k + 2\nu/L}.$$

where V^* is a minimum and $R(\mathbf{u}^0)$ is the diameter of $\{\mathbf{u} \in \mathbb{R}^n | V(\mathbf{u}) \le V(\mathbf{u}^0)\}$.

The next theorem concerns the convergence rate of Algorithm 5.2 when V is a PŁ-function, i.e. V satisfies the Polyak-Łojasiewicz inequality with parameter σ,

$$\frac{1}{2}\|\nabla V(\mathbf{u})\|^2 \le \sigma(V(\mathbf{u}) - V^*).$$

Note that under Assumption 5.1, all strongly convex functions are PŁ-functions [16].

Theorem 5.3. *If Assumption 5.1 holds and V is a PŁ-function, the iterates of Algorithm 5.2 satisfy, with ν as in Theorem 5.1,*

$$V(\mathbf{u}^k) - V^* \le \left(1 - \frac{2\sigma}{\nu}\right)^k (V(\mathbf{u}^0) - V^*).$$

Remark: The above theorems mean that for convex problems, too, the algorithm has a worst-case complexity of $\mathcal{O}(n^{1/2})$ with respect to the problem dimension n, compared to $\mathcal{O}(n^{3/2})$ for the cyclic coordinate descent algorithm [32] and $\mathcal{O}(n)$ for the expected bounds of stochastic coordinate descent [19]. We shall see in Lemma 5.1 that the complexity can be reduced further depending on a sparsity property of V.

5.3 The Euler's elastica problem

We will use Algorithm 5.1 for variational image analysis with Euler's elastica regularization. In variational image analysis one repairs a damaged input greyscale image $g : \Omega \to [0,1]$,where $\Omega \subset \mathbb{R}^2$ is often rectangular, by finding an output image $u : \Omega \to [0,1]$ that minimizes a functional

$$V_c(u) = d_c(K_c u, g) + \alpha J_c(u). \tag{5.3.1}$$

Here, K_c is a forward operator relating u to g, d_c a function measuring the distance between $K_c u$ and g, J_c a regularization functional and $\alpha > 0$ a constant. The subscript c emphasizes that the functions are continuous; they will later be discretized and renamed. When J_c is the Euler's elastica energy below, α is included in a and b.

The forward operator K_c, which may be linear, is inherent to the problem. For example, when considering an inpainting problem where the goal is to interpolate g in a subset D of the image domain Ω in which there is no given data, one would take K_c as a restriction to $\Omega \backslash D$. Since this leaves $K_c u$ undefined in Ω, the fidelity term should only compare with values of g on $\Omega \backslash D$. This has the effect of maintaining fidelity only in areas where the image is known, at the cost of generating an ill-posed problem due to non-unique solutions. In the denoising problem, where random noise is added to an image in unknown pixels, the usual choice is to take K_c as the identity operator since there is no information about which pixels are damaged.

The terms that differentiate approaches to image analysis are the d_c and J_c functions, and the implementation of the K_c operator if applicable. One often takes d_c as an L^p metric, while J_c can be chosen in several ways. A popular choice is the total variation (TV) [26] regularization which, for differentiable u, can be stated as

$$J_{TV}(u) = \int_\Omega |\nabla u| \mathrm{d}\mathbf{x}.$$

In practice, one often wishes to work with a differentiable function after dis-

cretizing J, thus using a smoothed version of J_{TV}, with $0 < \epsilon \ll 1$, given as

$$J_{TV_\epsilon}(u) = \int_\Omega |\nabla u|_\epsilon \mathrm{d}\mathbf{x} = \int_\Omega \sqrt{\frac{\partial u}{\partial x}^2 + \frac{\partial u}{\partial y}^2 + \epsilon}\,\mathrm{d}\mathbf{x}.$$

The Euler's elastica regularizer generalizes J_{TV}, adding a curvature dependent term. It is stated for $C^2(\Omega)$ functions u as [28]

$$J_c(u) = \int_\Omega \left(a + b\left(\nabla \cdot \frac{\nabla u}{|\nabla u|}\right)^2\right)|\nabla u|\mathrm{d}\mathbf{x},$$

where $a, b > 0$. We will consider the smoothed version

$$J_\epsilon(u) = \int_\Omega C(u)g(u)\mathrm{d}\mathbf{x}, \quad C(u) = a + b\left(\nabla \cdot \frac{\nabla u}{|\nabla u|_\epsilon}\right)^2, \quad g(u) = |\nabla u|_\epsilon. \quad (5.3.2)$$

where $C(u)$ and $g(u)$ are smoothed curvature and gradient terms.

5.3.1 Discretization

In the following, we assume that $\Omega = [1, n_x] \times [1, n_y]$, where n_x and n_y are the numbers of columns and rows, respectively, such that $n = n_x n_y$. We also assume that the image is in a pixel format, meaning we have input data $\mathbf{g}$ indexed as g_{ij} only at discrete points $\mathbf{x}_{ij} = (i, j)$. Thus, we must discretize (5.3.1) as

$$V(\mathbf{u}) = d(K\mathbf{u}, \mathbf{g}) + \alpha J(\mathbf{u}), \quad (5.3.3)$$

where $K : \mathbb{R}^n \to \mathbb{R}^n$ is a discretization of K_c, $\mathbf{u}$ is the output image indexed as u_{ij}, and d and J are discretizations of d_c and J_c. If d_c is an L^p norm, with K_c a restriction to $\Omega \backslash D$, we discretize it as

$$\left(\int_{\Omega\backslash D} |K_c u - g|^p \mathrm{d}\mathbf{x}\right)^{1/p} \approx \left(\sum_{(i,j)\in\Omega\backslash D} |(K\mathbf{u})_{ij} - g_{ij}|^p\right)^{1/p} =: d(K\mathbf{u}, \mathbf{g}).$$

If J_c is on integral form, one can use quadrature to discretize it as

$$J_c(u) = \int_\Omega H(u)\mathrm{d}\mathbf{x} \approx \sum_{i,j} H(u)\big|_{\mathbf{x}_{ij}}.$$

Since it requires derivatives of u, $H(u) = C(u)g(u)$ in (5.3.2) must be approximated at the points $\mathbf{x}_{ij}$ by values $H_{ij}(\mathbf{u})$, such that the final discretization becomes

$$J_c(u) \approx \sum_{i,j} H_{ij}(\mathbf{u}).$$

For this approximation we use finite differences on a staggered grid as in [28] and [30]. The stencil used for discretizing both $g(u)$ and $C(u)$ is shown in Figure 5.1. The u_{ij} are shown as green squares, and u_x and u_y are approximated by finite differences at red and blue points. With these, we approximate $g(u)$ and $C(u)$. Following the standard approach for TV regularization, $g(u)$ is ap-

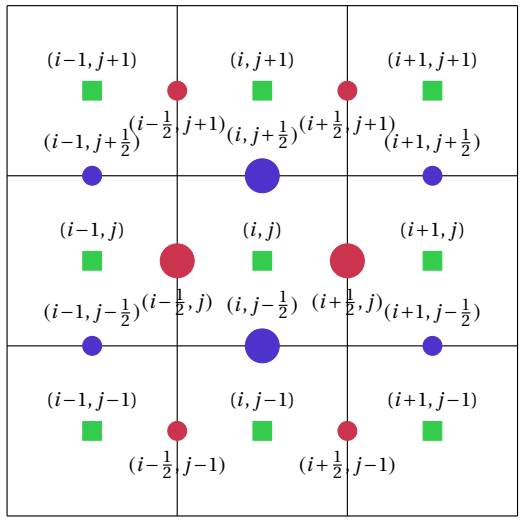


Figure 5.1: Discretization stencil. Green squares: Pixel data u_{ij}. All red/blue circles: approximations of u_x and u_y. Large red/blue circles: approximations of x and y components of $\nabla u/|\nabla u|_\epsilon$.

proximated by backward differences. Approximating $C(u)$ requires evaluation of the x and y components of $\frac{\nabla u}{|\nabla u|_\epsilon}$ at the large dots (red for the x component, blue for the y component) and taking central differences of these to approximate the divergence. To evaluate $|\nabla u|_\epsilon$, we approximate values for u_y at the large red dots and u_x at the large blue dots by the mean of the u_y and u_x approximations at the four nearest blue and red points, respectively. In total, the discretized regularizer is

$$J(\mathbf{u}) = \sum_{i=1}^{n_x}\sum_{j=1}^{n_y}\left(a + b\left(\delta_x^+\frac{\delta_x^- u_{ij}}{w_{i-\frac{1}{2},j}} + \delta_y^+\frac{\delta_y^- u_{ij}}{w_{i,j-\frac{1}{2}}}\right)^2\right) g_{ij}. \tag{5.3.4}$$

Here, δ_x^+, δ_x^-, δ_y^+, and δ_y^- denote forward/backward differences in x and y directions,

$$\delta_x^+ f_{ij} = f_{i+1,j} - f_{ij}, \qquad \delta_x^- f_{ij} = f_{ij} - f_{i-1,j},$$
$$\delta_y^+ f_{ij} = f_{i,j+1} - f_{ij}, \qquad \delta_y^- f_{ij} = f_{ij} - f_{i,j-1},$$

and the discretization of the $|\nabla u|_\epsilon$ terms depends on the point as

$$g_{ij} = \sqrt{(\delta_x^- u_{ij})^2 + (\delta_y^- u_{ij})^2 + \epsilon},$$
$$w_{i-\frac{1}{2},j} = \sqrt{(\delta_x^- u_{ij})^2 + (\delta_y^* u_{ij})^2 + \epsilon},$$
$$w_{i,j-\frac{1}{2}} = \sqrt{(\delta_x^* u_{ij})^2 + (\delta_y^- u_{ij})^2 + \epsilon},$$

where

$$\delta_x^* u_{ij} = \frac{1}{4}(\delta_x^- u_{i+1,j} + \delta_x^- u_{ij} + \delta_x^- u_{i+1,j-1} + \delta_x^- u_{i,j-1})$$
$$\delta_y^* u_{ij} = \frac{1}{4}(\delta_y^- u_{i,j+1} + \delta_y^- u_{ij} + \delta_y^- u_{i-1,j} + \delta_y^- u_{i-1,j+1}).$$

The discrete energy (5.3.4) has a Lipschitz continuous gradient, where the Lipschitz constant depends on ϵ. Thus, any energy of the form (5.3.3) using (5.3.4) as a regularizer will satisfy Theorem 5.1 when the fidelity term has a Lipschitz continuous gradient.

Figure 5.2: Image split into $M = 8$ blocks B_m, $1 \le m \le 8$, with cyan border sets $\Gamma_l, 1 \le l \le 7$.

5.3.2 Decoupling and parallelization

Algorithms 5.1 and 5.2 follow a cyclic ordering with elements updated column-wise, but Theorems 5.1, 5.2 and 5.3 make no assumptions on element ordering, so the convergence rates are unaffected by reordering updates. Also, Figure 5.1 indicates that updating u_{ij} will only affect the $H_{\mu\nu}(\mathbf{u})$ with (μ, ν) immediately surrounding (i, j). Hence, if we split the image in M parts as shown in Figure 5.2 and sweep through the cyan elements first, the blocks separated by cyan pixels can be updated independently of each other and thus in parallel, a domain decomposition strategy similar to that in [33].

Algorithm 5.3. DG-PARALLEL

Choose $\tau > 0$, $tol > 0$ and $\mathbf{u}^0 \in \mathbb{R}^n$. Set $k = 0$. Initialize M threads.
Define M index sets B_m and N index sets Γ_l.
repeat
parallel: N threads. Thread number l does:
$\mathbf{v}_0^{k,l} = \mathbf{u}^k$
for $j \in \Gamma_l$ **do**
Solve $\beta_j^k = -\tau(V(\mathbf{v}_{j-1}^{k,l} + \beta_j^k \mathbf{e}_j) - V(\mathbf{v}_{j-1}^{k,l}))/\beta_j^k$
$\mathbf{v}_j^{k,l} = \mathbf{v}_{j-1}^{k,l} + \beta_j^k \mathbf{e}_j$
end for
Reduce: $\mathbf{v}_0^k = \mathbf{u}^k + \sum_{j \in \cup_{l=1}^N \Gamma_l} \beta_j^k \mathbf{e}_j$
Parallel: M threads. Thread number m does:
$\mathbf{v}_0^{k,m} = \mathbf{v}_0^k$
for $j \in B_m$ **do**
Solve $\beta_j^k = -\tau(V(\mathbf{v}_{j-1}^{k,m} + \beta_j^k \mathbf{e}_j) - V(\mathbf{v}_{j-1}^{k,m}))/\beta_j^k$
$\mathbf{v}_j^{k,m} = \mathbf{v}_{j-1}^{k,m} + \beta_j^k \mathbf{e}_j$
end for
Reduce: $\mathbf{u}^{k+1} = \mathbf{v}_0^k + \sum_{j \in \cup_{m=1}^M B_m} \beta_j^k \mathbf{e}_j$
$k = k + 1$
until $\left(V(\mathbf{u}^k) - V(\mathbf{u}^{k-1})\right)/V(\mathbf{u}^0) < tol$

This inspires Algorithm 5.3, a parallel version of Algorithm 5.1 where the indices of the unknowns are divided into two collections of index sets, $\{B_m\}_{m=1}^M$ and $\{\Gamma_l\}_{l=1}^N$, based on the dependency radius of V, defined below. Note that the acceleration procedure proposed in Algorithm 5.2 still works here. For a rigorous discussion, we first introduce a distance measure between index sets. Define the distance between two index pairs (i, j) and (k, l) by

$$\mathrm{dist}_{\mathrm{ind}}\big((i,j),(k,l)\big) = \max\{|i-k|, |j-l|\},$$

which is the graph distance when every index has edges to the closest indices vertically, horizontally and diagonally. We define the distance between two index sets as

$$\mathrm{dist}_{\mathrm{set}}(I_m, I_n) = \min_{\substack{(i,j)\in I_m \\ (k,l)\in I_n}} \mathrm{dist}_{\mathrm{ind}}\big((i,j),(k,l)\big).$$

If $\mathrm{dist}_{\mathrm{set}}(I_m, I_n) = 0$, then I_m and I_n share at least one index; if $\mathrm{dist}_{\mathrm{set}}(I_m, I_n) = 1$, at least one index in I_m is adjacent to an index in I_n, horizontally, vertically or diagonally; if $\mathrm{dist}_{\mathrm{set}}(I_m, I_n) = 2$, there is a band of width 1 of indices separating I_m and I_n, et cetera. We can now define the dependency radius of a function;

we say that $V : \mathbb{R}^n \to \mathbb{R}$ has dependency radius R if for all $\delta \in \mathbb{R}$ and (i, j),

$$V(\mathbf{u} + (\delta - u_{ij})\mathbf{e}_{ij}) - V(\mathbf{u}) = F(\mathbf{u}_{ij}^R(\mathbf{u}), \delta),$$

where $F : \mathbb{R}^{(2R+1)^2+1} \to \mathbb{R}$ is a function depending on δ and

$$\mathbf{u}_{ij}^R(\mathbf{u}) = (u_{i_{\min}, j_{\min}}, ..., u_{ij}, ..., u_{i_{\max}, j_{\max}}),$$

where

$$\begin{aligned} i_{\min} &= \max\{i - R, 1\}, & j_{\min} &= \max\{j - R, 1\} \\ i_{\max} &= \min\{i + R, n_x\}, & j_{\max} &= \min\{j + R, n_y\}. \end{aligned}$$

This means that computing the change in V from updating unknown number (i, j) requires only the unknowns with indices within a distance of R. In the discretized Euler's elastica problem we have $R = 1$. If V has dependence radius R, then u_{ij} can be updated using the Itoh-Abe discrete gradient independently of u_{kl} if $\text{dist}_{\text{ind}}((i, j), (k, l)) > R$. We can decouple a problem with dependency radius R by choosing M index sets $B_m \subset \Omega$ such that $\text{dist}_{\text{set}}(B_m, B_n) > R$ for all (m, n). Then, one chooses a second collection of N index sets Γ_l such that $\text{dist}_{\text{set}}(\Gamma_k, \Gamma_l) > R$ and $\cup_{l=1}^N \Gamma_l = \Omega / \cup_{m=1}^M B_m$. Note that in general, $M \neq N$ and that while this discussion has been focused on two-dimensional indexing, generalizing dist_{ind} and dist_{set} in the obvious manner to higher-dimensional index pairs admits a similar approach in arbitrary indexing dimensions.

5.3.3 Effect of dependency radius on complexity of the algorithm

A consequence of V having dependency radius R is that $\partial V / \partial u_{ij}$ depends on $\mathbf{u}_{ij}^R$ only. This can be used to make sharper versions of Theorems 5.1, 5.2 and 5.3 through a property presented in the following lemma for two-dimensional indexing.

Lemma 5.1. *If Assumption 5.1 holds and in addition V has dependency radius R, the $\mathbf{u}^k$ produced by Algorithm 5.2 satisfy*

$$\|\nabla V(\mathbf{u}^k)\|^2 \leq \nu \left(V(\mathbf{u}^k) - V(\mathbf{u}^{k+1}) \right), \tag{5.3.5}$$

with

$$\nu = 2L_{\max}^2 \left((2R + 1)^2 \tau_{\max} + \frac{\tau_{\max}}{L_{\max}^2 \tau_{\min}^2} \right).$$

Proof. Recall the coordinatewise formulation of the Itoh–Abe scheme, with 2D indexing where, with $\beta_{lm}^k = u_{lm}^{k+1} - u_{lm}^k$,

$$\beta_{lm}^k = -\tau_k \frac{V\left(\mathbf{v}_{l,m}^k\right) - V\left(\mathbf{v}_{l,m-1}^k\right)}{\beta_{lm}^k}.$$

Here, $\mathbf{v}_{l,m}^k := \mathbf{u}^k + \sum_{i=1}^{l-1}\sum_{j=1}^{n_y}(\beta_{ij}^k)\mathbf{e}_{ij} + \sum_{j=1}^{m}(\beta_{lj}^k)\mathbf{e}_{lj}$. We follow the proof of Theorem 5.1 up to (5.2.7), where we exploit the dependence radius R of V. For an $s \in (0,1)$, we have

$$\begin{aligned}\left|\frac{\partial V}{\partial u_{lm}}(\mathbf{u}^{k+1})\right| &= \left|\frac{\partial V}{\partial u_{lm}}(\mathbf{u}_{lm}^R(\mathbf{u}^{k+1}))\right| \\ &\leq \left|\frac{\partial V}{\partial u_{lm}}(\mathbf{u}_{lm}^R(\mathbf{u}^{k+1})) - \frac{\partial V}{\partial u_{lm}}(\mathbf{u}_{lm}^R(\mathbf{v}_{l,m-1}^k) + s\beta_{lm}^k\mathbf{e}_{lm})\right| + \frac{\left|\beta_{lm}^k\right|}{\tau_k}.\end{aligned}$$

Using coordinatewise Lipschitz continuity, we have

$$\left|\frac{\partial V}{\partial u_{lm}}(\mathbf{u}^{k+1})\right| \leq L_{lm}\|\mathbf{u}_{lm}^R(\mathbf{u}^{k+1}) - \mathbf{u}_{lm}^R(\mathbf{v}_{l,m-1}^k) - s\beta_{lm}^k\mathbf{e}_{lm}\| + \frac{\left|\beta_{lm}^k\right|}{\tau_k},$$

and summing up over all coordinates, we get

$$\begin{aligned}\|\nabla V(\mathbf{u}^{k+1})\|^2 &\leq \sum_{l=1}^{n_x}\sum_{m=1}^{n_y}\left(L_{lm}\|\mathbf{u}_{lm}^R(\mathbf{u}^{k+1}) - \mathbf{u}_{lm}^R(\mathbf{v}_{l,m-1}^k) - s\beta_{lm}^k\mathbf{e}_{lm}\| + \frac{\left|\beta_{lm}^k\right|}{\tau_k}\right)^2 \\ &\leq 2\sum_{l=1}^{n_x}\sum_{m=1}^{n_y}\left(L_{lm}^2\left(\sum_{i=l_{\min}}^{l_{\max}}\sum_{j=m_{\min}}^{m_{\max}}{\beta_{ij}^k}^2\right) + \frac{\left|\beta_{lm}^k\right|^2}{\tau_k^2}\right) \\ &\leq 2L_{\max}^2\left((2R+1)^2 + \frac{1}{L_{\max}^2\tau_{\min}^2}\right)\|\mathbf{u}^{k+1} - \mathbf{u}^k\|^2.\end{aligned}$$

Since $\|\mathbf{u}^{k+1} - \mathbf{u}^k\|^2 = \tau_k(V(\mathbf{u}^k) - V(\mathbf{u}^{k+1}))$, this concludes the proof. □

Remark: This improved estimate affects the complexity of Theorems 5.1, 5.2 and 5.3, reducing it from a worst-case of $\mathcal{O}(n^{1/2})$ to $\mathcal{O}((2R+1)^2)$ for convex two-dimensionally indexed problems. Indeed, choosing a constant step size τ minimizing ν, one obtains

$$\tau = \frac{1}{(2R+1)L_{\max}}, \qquad \nu = 4(2R+1)L_{\max},$$

meaning the complexity scales as $\mathcal{O}(R)$. This is of particular interest for problems involving discretizations such as the Euler's elastica regularization considered here. For general D-dimensional indexing one can expect the complexity to scale as $\mathcal{O}((2R+1)^D)$, with improvements from optimal step size selection.

5.4 Numerical experiments

In this section, we first apply Euler's elastica regularization to denoising problems and to image inpainting. The results are compared to those of TV regularization to verify the qualitative improvement of Euler's elastica regularization with Algorithm 5.1 over TV. This is not intended as an account on the competitiveness of Euler's elastica against other regularising procedures in general but serves as a proof of concept for the qualitative and algorithmic performance of the discrete gradient approach for Euler elastica when compared to discrete gradient for TV. We further investigate the convergence rate numerically, with varying smoothing constant ϵ and ordering of the unknowns. We also verify the convergence rate's independence of n as predicted by Lemma 5.1. Next, we compare the execution time to another state-of-the-art algorithm for non-convex optimization, the iPiano algorithm [23], and to the gradient descent and Heavy-ball algorithms. Finally, we evaluate the algorithm's sensitivity to the initial guess.

All algorithms were implemented as hybrid MATLAB and C functions using the MATLAB EXecutable (MEX) interface, where critical parts of the code are implemented in C. The tests were executed using MATLAB (2017a release) running on a Mid 2014 MacBook Pro with a four-core 2.5 GHz Intel Core i7 processor and 16 GB of 1600 MHz DDR3 RAM. For the Brent-Dekker algorithm implementation we used the built-in MATLAB function `fzero`, and block parallelization was done using MATLAB's `blockproc` and `parfor` functions.

5.4.1 Image denoising

We first consider denoising images. The typical choice of fidelity term is an L^p metric where p depends on the type of noise encountered. The discretized forward operator K is the identity operator. We wish to minimize

$$V(\mathbf{u}) = \sum_{i,j} |u_{ij} - g_{ij}|^p + J(\mathbf{u}). \tag{5.4.1}$$

In the first example we have added Gaussian noise with a standard deviation of 0.2, using $p = 2$ for the fidelity term. In the second example we have added impulse noise, randomly setting the values of 25% of the pixels to either 0 or 1 as depicted in the top pictures in Figure 5.4. Impulse noise is removed using $p = 1$ in (5.4.1). Note that the fidelity term with $p = 1$ is non-differentiable and falls outside of the theoretical basis of section 5.2.2. It is still included here to see how the method behaves beyond the smooth setting.

Figure 5.3 shows Euler's elastica denoising with $p = 2$ applied to an image corrupted by Gaussian noise in its lower right hand panel and a TV regularized version in the lower left hand panel. For both TV and elastica denoising, we

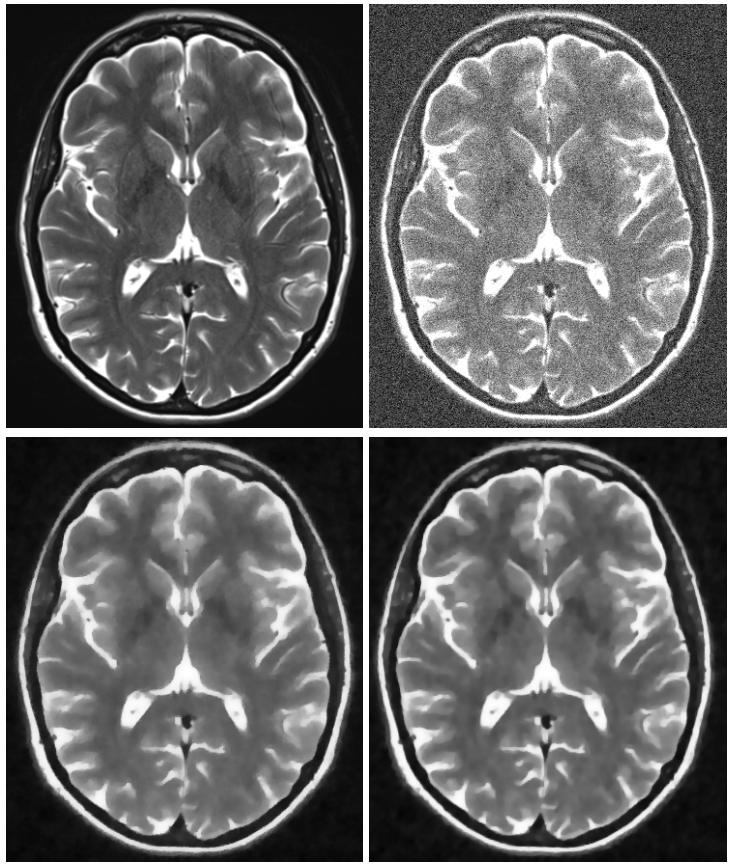


Figure 5.3: Denoising with $p = 2$. Top left: original image. Top right: noisy input image g. Bottom left: TV denoised. PSNR: 20.9421, SSIM: 0.8398. Bottom right: Elastica denoised. PSNR: 21.3760, SSIM: 0.8595.

chose $\epsilon = 10^{-4}$; larger values resulted in blurring and lower values showed no visible improvement but slower convergence. For TV denoising, we set $a = 0.17$, and for elastica denosing, we chose $a = 0.9$ and $b = 0.9$. These values were chosen to maximize PSNR and SSIM.

In the lower right hand panel of Figure 5.4, the result of elastica denoising with $p = 1$ on a picture of the Louvre, corrupted by impulse noise, is shown. A TV regularized version is seen in the lower left hand panel. As above, we chose $\epsilon = 10^{-4}$ for both TV and elastica denoising. In the TV case, we set $a = 0.8$, and in the elastica case, we chose $a = 0.4$ and $b = 0.2$. These values were chosen to maximize PSNR and SSIM.

As can be seen in both examples, the results of Euler's elastica denoising are slightly more visually appealing than the TV denoised versions, which is as expected since Euler's elastica generalizes TV regularization. In Figure 5.3,

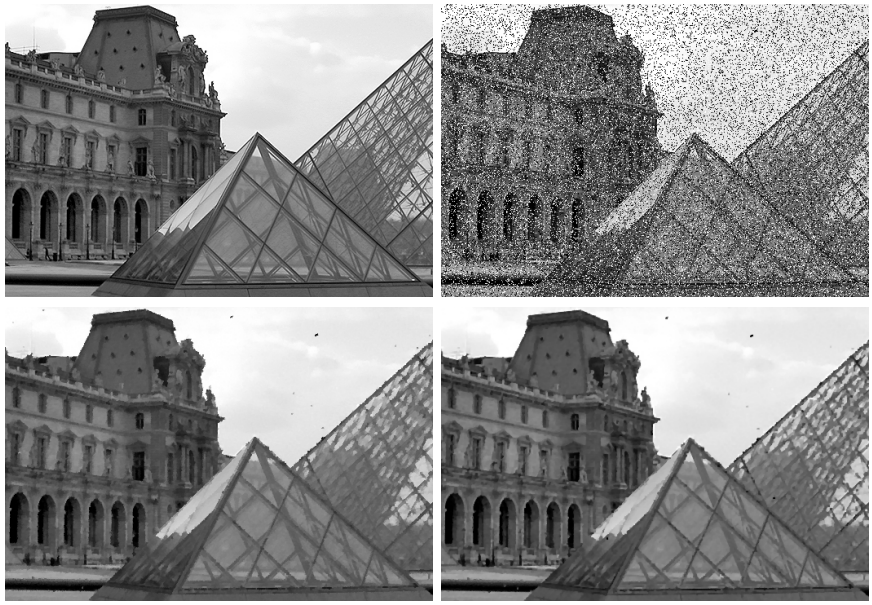


Figure 5.4: Denoising with $p = 1$. Left: TV denoised. PSNR: 25.7287, SSIM: 0.8572. Right: Elastica denoised. PSNR: 25.9611, SSIM: 0.8628.

edges are sharper and the contrast level better in the elastica denoised image. In Figure 5.4 the pyramids' lines are sharper and the museum's façade details are clearer. One can also see that the textures are smoother and that edges are less jagged in the elastica reconstruction.

5.4.2 Image inpainting

Inpainting is used when there is data loss in known pixels. Using $p = 2$ and a discretized restriction operator K we wish to minimize

$$V(\mathbf{u}) = \sum_{(i,j)\in\Omega\setminus D} (u_{ij} - g_{ij})^2 + J(\mathbf{u}),$$

where $D \subset \Omega$ is the damaged domain. Figure 5.5 shows the result of applying Algorithm 5.1 to an example inpainting problem. Here, the top left panel shows the original image, the top right panel the image with 95% of the pixels removed randomly, the bottom left panel a TV inpainted image, and the bottom right panel an Euler's elastica inpainted image. For both TV and elastica, we chose $\epsilon = 10^{-4}$. In the TV case, we set $a = 2.5 \cdot 10^{-7}$, and in the elastica case, we chose $a = 10^{-6}$ and $b = 10^{-5}$. These values were chosen to maximize SSIM. Here, the superiority of Euler's elastica is evident, as more details are reconstructed and the image appears sharper than with TV regularization.

Figure 5.5: Top left: Original image. Top right: 95 % random data loss. Bottom left: Inpainted with TV - SSIM: 0.7512. Bottom right: Inpainted with elastica - SSIM: 0.8896.

5.4.3 Convergence rates

When denoising using $p = 2$ in (5.4.1), the conditions of Theorem 5.1 are fulfilled, and so we investigate the convergence rates numerically in this case. The two plots in Figure 5.6 show $\min_{0\leq l\leq k} \|\nabla V(\mathbf{u}^l)\|/\|\nabla V(\mathbf{u}^0)\|$ for DG and DG-ADAPT applied to the Euler's elastica regularized denoising problem with $p = 2$ shown in Figure 5.3 for two choices of ϵ. Each plot shows the result of initializing with a τ_0 chosen by trial and error to yield the best convergence rate, and with a much smaller τ_0. Both algorithms were started from the same random initalization $\mathbf{u}^0$ and with the same initial time step τ_0. For DG-ADAPT, the additional parameters were chosen as $\rho = 0.99$, $c_1 = 0.7, c_2 = 0.9$ and $\gamma = 1.005$. Note that the left hand plot is semilogarithmic while the right hand plot is logarithmic. The left hand plot shows linear convergence for the DG algorithm when τ_0 is chosen correctly and a much slower rate for the suboptimal τ_0. The linear convergence can be expected by Theorem 5.3 if the choice of $\epsilon = 10^{-4}$ means V, which is twice differentiable, becomes strongly convex in a neigh-

bourhood of the minimize, or if it is a PŁ-function. In the right hand plot, where $\epsilon = 10^{-7}$ leads to a more ill-conditioned problem, the convergence rate appears closer to that predicted in Theorem 5.2, indicating that a neighborhood of strong convexity has not yet been reached. Both plots show that using adaptive step sizes yields faster convergence than fixed step sizes, especially when τ_0 is not carefully chosen beforehand.

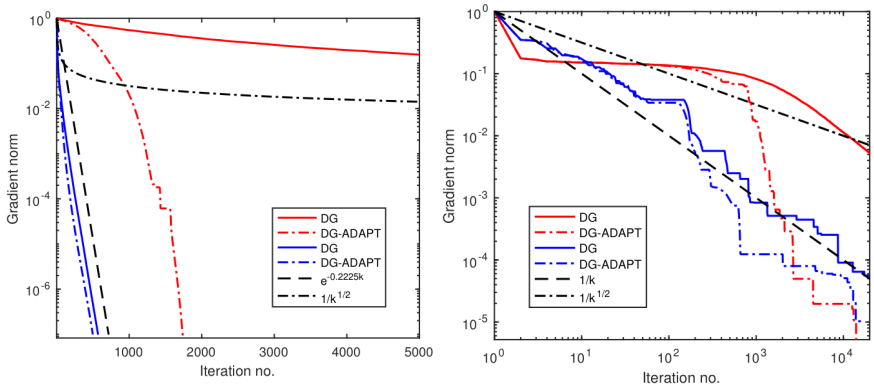


Figure 5.6: Convergence rates in terms of $\min_{0\leq l\leq k} \|\nabla V(\mathbf{u}^l)\|/\|\nabla V(\mathbf{u}^0)\|$ for the Euler's elastica regularized denoising problem with $p = 2$ of illustrated in Figure 5.3. Blue denotes $\tau_0 = 0.38$, red denotes $\tau_0 = 0.38 \cdot 10^{-4}$. Left: Using $\epsilon = 10^{-4}$. Right: Using $\epsilon = 10^{-7}$.

Figure 5.7 shows convergence rates for four different element orderings. The first ordering is the natural ordering which iterates over pixels starting in one corner and proceeding columnwise. The second is red-black ordering where pixels u_{ij} with $i + j$ even are updated first, then pixels with $i + j$ odd. Third is a random ordering, with the same ordering used for all time steps. Last, we consider the block ordering of the parallelized algorithm as illustrated in Figure 5.2. The plots of Figure 5.7 concern the same problem as Figure 5.6, but with the DG algorithm only and showing rates in terms of the relative optimality error $(V(\mathbf{u}^k) - V^*)/(V(u^0) - V^*)$, where V^* was produced by running the algorithm for 20 000 iterations. Step sizes were chosen individually for the different orderings to produce the best possible convergence. The left hand plot plateaus at a relative optimality error of $\sim 10^{-16}$, i.e. machine precision. Both plots show that the asymptotic convergence rate, represented by the slope of the lines, is similar for all orderings, as is to be expected from the independency of ordering in the convergence theorems. However, the early rate of the random and red-black orderings of the left hand plot is better than that of the natural and block orderings, suggesting that the choice of ordering affects the constant ν in Theorems 5.1, 5.2 and 5.3. We consider this an interesting topic for further investigation.

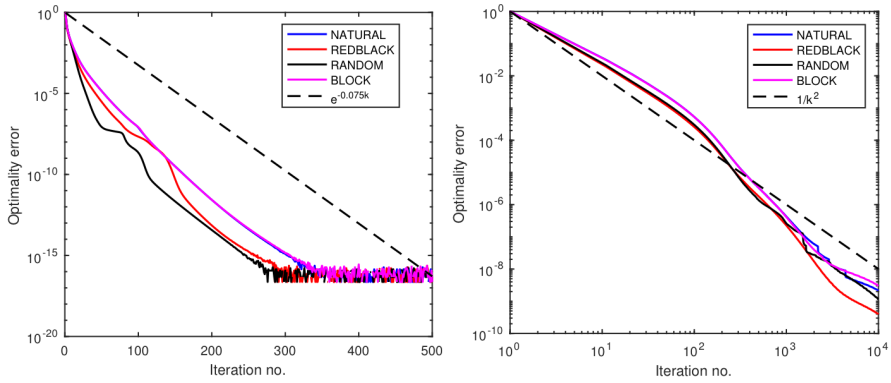


Figure 5.7: Convergence rates in terms of $(V(\mathbf{u}^k) - V^*)/(V(\mathbf{u}^0) - V^*)$ for the Euler's elastica regularized denoising problem with $p = 2$ illustrated in Figure 5.3, with different orderings. Left: Using $\epsilon = 10^{-4}$. Right: Using $\epsilon = 10^{-7}$.

Figure 5.8 shows convergence rates in terms of $(V(\mathbf{u}^k) - V^*)/(V(\mathbf{u}^0 - V^*)$ for the DG algorithm applied to the denoising problem (5.4.1) with $p = 1$. Here, all parameters are fixed but the input image is rescaled to $2^k \times 2^k$, $k = 4, 5, 6, 7, 8, 9$, before being subjected to impulse noise, randomly setting values of 25% of the pixels to 0 or 1 before denoising. The plot verifies the conclusion of Lemma 5.1, that the asymptotic convergence rates of Algorithms 5.1 and 5.2 are independent of the problem size n when the objective function has sparsely connected unknowns.

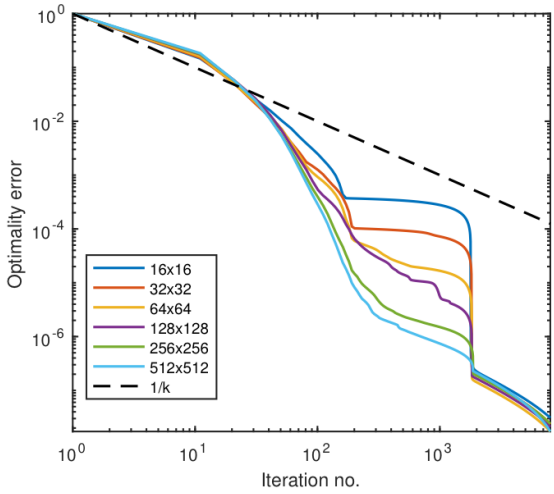


Figure 5.8: Convergence rates in terms of $(V(\mathbf{u}^k) - V^*)/(V(\mathbf{u}^0) - V^*)$ for the Euler's elastica regularized denoising problem with $p = 1$ illustrated in Figure 5.4, for varying problem sizes.

5.4.4 Execution time

As a general algorithm suited to the kind of non-convex minimization problems that the Euler's elastica problem poses, it is reasonable to compare the DG algorithms to the iPiano algorithm of [23]; in particular, we use Algorithm 4 from this article. Inspired by Polyak's Heavy-ball algorithm and the proximal gradient algorithm, iPiano considers minimization problems of the form

$$\min_{\mathbf{u}\in\mathbb{R}^n} f(\mathbf{u}) + g(\mathbf{u}),$$

where g is convex and possibly non-smooth while f is smooth and possibly non-convex, and iterates based on the update scheme

$$\mathbf{u}^{k+1} = (I + \alpha\partial g)^{-1}(\mathbf{u}^k - \alpha\nabla f(\mathbf{u}^k) + \beta(\mathbf{u}^k - \mathbf{u}^{k-1})),$$

where $(I + \alpha\partial g)^{-1}$ denotes a proximal step by

$$(I + \alpha\partial g)^{-1}(\mathbf{y}) = \arg\min_{\mathbf{z}\in\mathbb{R}^n} \frac{\|\mathbf{z}-\mathbf{y}\|^2}{2} + \alpha g(\mathbf{z}).$$

We wish to time the algorithms on a problem with non-differentiable terms, and so we use a variation on the discrete elastica regularizer (5.3.4), taking

$$J(\mathbf{u}) = a\sum_{i=1}^{n_x}\sum_{j=1}^{n_y} \bar{g}_{ij} + b\sum_{i=1}^{n_x}\sum_{j=1}^{n_y}\left(\delta_x^+\frac{\delta_x^- u_{ij}}{w_{i-\frac{1}{2},j}} + \delta_y^+\frac{\delta_y^- u_{ij}}{w_{i,j-\frac{1}{2}}}\right)^2 g_{ij} =: aT(\mathbf{u}) + bK(\mathbf{u}),$$

where

$$\bar{g}_{ij} = \sqrt{(\delta_x^- u_{ij})^2 + (\delta_y^- u_{ij})^2}.$$

That is, the TV term T is not differentiable but the curvature term K is. The choice of f and g in the iPiano algorithm depends on whether the fidelity term is differentiable or not. We take $f = K + d$ if d is the discrete L^2 fidelity term, but $f = K$ if d is the discrete L^1 fidelity term. Likewise, we take $g = T$ if d is the discrete L^2 fidelity term, but $g = T + d$ if d is the discrete L^1 fidelity term. In both cases, evaluating $(I + \alpha\partial g)^{-1}$ is equivalent to solving a TV regularization problem, which is done efficiently using the Chambolle-Pock algorithm [8]. This algorithm can be accelerated in the case of a uniformly convex fidelity term, i.e. if d is a discrete L^2 norm. If it is not, as is the case when d is a discrete L^1 norm, no acceleration is possible. The DG-ADAPT algorithm requires the computation of gradients; they are computed using the smoothed (5.3.4); also, when using the non-differentiable L^1 fidelity term, the additional smoothing

$$\|u-g\|_{L^1} = \sum_{i,j}|u_{ij}-g_{ij}| \approx \sum_{i,j}\sqrt{(u_{ij}-g_{ij})^2+\varepsilon} := \|u-g\|_{L^1,\varepsilon}$$

was used with $\varepsilon = 10^{-12}$ to compute gradients in the DG-ADAPT algorithm.

All algorithms were implemented in serial versions using MEX, and timed. Note that the above non-smoothed TV term was used in the DG/DG-ADAPT algorithms as well for this test. Tables 5.1 and 5.2 show timing results for a denoising test on a 512×512 image for different values of ϵ using an L^2 norm and an L^1 norm for the fidelity term, respectively. For each ϵ, a reference solution $\bar{V}$ was found by running the DG algorithm for 20 000 iterations or until a minimizer was found with machine precision. The algorithms were then tested on the problem, running until the iterations reached a value of $V(\mathbf{u}^k) \leq 1.0001 \cdot \bar{V}$ or 4000 iterations. Both algorithms require the solution of a subproblem; either a root finding problem for the DG algorithms, or the evaluation of a proximal operator for the iPiano algorithm. For the DG algorithms the tolerance of the root finding algorithm was kept at a fixed value, while for the iPiano algorithm, the tolerance in the prox operator evaluation was adjusted to obtain the fastest runtime while still converging. In the DG-ADAPT algorithm the parameter choices $c_1 = 0.7, c_2 = 0.9, \rho = 0.98$ and $\gamma = 1.005$ were used for the L^2 test, and $c_1 = 0.2, c_2 = 0.7, \rho = 0.995$ and $\gamma = 1.0025$ were used for the L^1 test.

Table 5.1: Results of L^2 test. Format: (Iterations/CPU time (s)). Best times in bold.

ϵ	iPiano	DG	DG-ADAPT
10^{-1}	**30/8.90**	29/15.61	28/17.39
10^{-2}	**32/9.60**	28/14.79	27/16.54
10^{-3}	**38/12.40**	38/20.42	35/21.67
10^{-4}	**59/16.78**	67/38.22	57/36.62
10^{-5}	**216/47.74**	115/69.51	89/60.20
10^{-6}	1684/556.11	180/115.26	**131/91.19**
10^{-7}	3968/1071.37	269/196.12	**204/151.71**

From both tables, it is apparent that the DG and DG-ADAPT algorithms both scale better with ϵ than iPiano, which reached the maximum number of iterations at $\epsilon = 10^{-6}$ in the L^1 test and hence was not timed with $\epsilon = 10^{-7}$. However, for larger values of ϵ, iPiano appears to be the better choice. The time usage per iteration increases with ϵ for both iPiano and DG/DG-ADAPT; for iPiano, this is due to the precision in the prox operator evaluation increasing which requires more time. For DG/DG-ADAPT, the slight increase can be explained by an increase in the amount of iterations needed by the Brent-Dekker algorithm to solve the scalar subproblems. Also note that in Table 5.2, we can see that DG-ADAPT is not noticeably faster than DG in most cases, indicat-

Table 5.2: Results of L^1 test. Format: (Iterations/CPU time (s)). Best times in bold.

ϵ	iPiano	DG	DG-ADAPT
10^{-1}	**23/21.73**	199/171.82	168/161.73
10^{-2}	**59/29.67**	288/250.23	247/231.56
10^{-3}	**146/46.36**	305/255.12	252/231.50
10^{-4}	**401/229.04**	300/246.27	253/223.21
10^{-5}	1399/2181.81	**303/246.90**	292/257.47
10^{-6}	4000/18566.24	**303/242.00**	304/267.11
10^{-7}	N/A	**400/335.10**	423/373.91

ing that using gradients of smoothed versions of the objective function is not sufficient to accelerate convergence for non-smooth problems.

Table 5.3: Results of L^2 test with smoothed TV term. Format: (Iterations/CPU time (s)). Best times in bold.

ϵ	Gradient Descent	Heavy-ball	DG	DG-ADAPT
10^{-1}	**12/5.16**	23/12.79	20/9.46	19/8.61
10^{-2}	27/14.78	29/17.95	25/12.72	**24/11.85**
10^{-3}	145/101.00	38/25.16	32/17.43	**30/15.76**
10^{-4}	388/319.57	96/73.59	**42/24.66**	43/24.71
10^{-5}	1310/1271.73	312/284.49	**68/43.95**	76/47.92
10^{-6}	4001/4621.38	1018/1079.10	119/83.15	**112/73.07**
10^{-7}	N/A	2922/3765.52	194/142.05	**181/116.88**
10^{-8}	N/A	4001/5488.83	376/291.73	**398/260.18**

Table 5.3 shows timing results of denoising test problems with the smoothed elastica regularizer (5.3.4), on a 512×512 image using a squared L^2 fidelity term, comparing the DG and DG-ADAPT algorithms to the gradient descent and Heavy-ball algorithms with Armijo step size selection. Here, the parameters of the DG-ADAPT algorithm were chosen as $\rho = 0.99$, $c_1 = 0.7$, $c_2 = 0.9$ and $\gamma = 1.005$ after experimentation. The τ parameter in the DG algorithm was chosen to give the fastest convergence at $\epsilon = 10^{-4}$, and the same τ was used as initial τ_0 in DG-ADAPT. The table shows that the DG and DG-ADAPT algorithms outperform the gradient descent and Heavy-ball algorithms on the problem for all ϵ except $\epsilon = 10^{-1}$, with increasing difference as $\epsilon \to 0$. The gradient descent algorithm reached the maximum of 4000 iterations at $\epsilon = 10^{-6}$ and was therefore not tested with smaller ϵ.

Using the DG-PARALLEL algorithm with column splitting, a speedup of 2-2.5 was observed when employing 4 cores, with larger speedup values for larger images.

5.4.5 Dependence on starting point

To investigate the influence of the starting point on the performance of the algorithm in the minimisation of the non-convex Euler elastica problem, we tested the inpainting problem with DG and iPiano, comparing execution time and reconstruction quality of the two algorithms when starting from a random starting image, a unicolor (black) starting image, or from the original image prior to removing pixels.

As seen in Figure 5.10, the DG algorithm produces different but acceptable reconstructions depending on the starting guess. The iPiano algorithm works with a random initialization and when starting from the original image, but is worse with a unicolor starting image. It is reasonable to expect that starting from the original image gives the best reconstruction, and so we compare reconstructions starting from other images to this. Figure 5.9 shows the differences between image obtained when starting from the original and the images obtained when starting from either random or unicolor images. We can see that the reconstructions are largely in agreement except in certain areas such as the stamens of the top right flower. In Table 5.4 we see that the iPiano algorithm is slower than the DG algorithms when it comes to inpainting, and also that the unicolor initialization produced an answer which was pretty far from optimal as compared to the other initializations.

Table 5.4: Results of inpainting test. Format: (Final energy/CPU time (s)). Best times in bold.

Initialization	iPiano	DG	DG-ADAPT
Random	0.012334/3434	**0.012323/431**	0.012323/460
Unicolor	0.014892/2934	0.012324/2740	**0.012324/438**
Original	0.012263/1131	**0.012263/145**	0.012263/208

5.5 Conclusion

We have introduced a novel method for solving non-convex optimization problems and tested it on Euler's elastica regularized variational image analysis problems. We have produced a convergence rate estimate for non-convex problems assuming that V is continuously differentiable with Lipschitz continuous

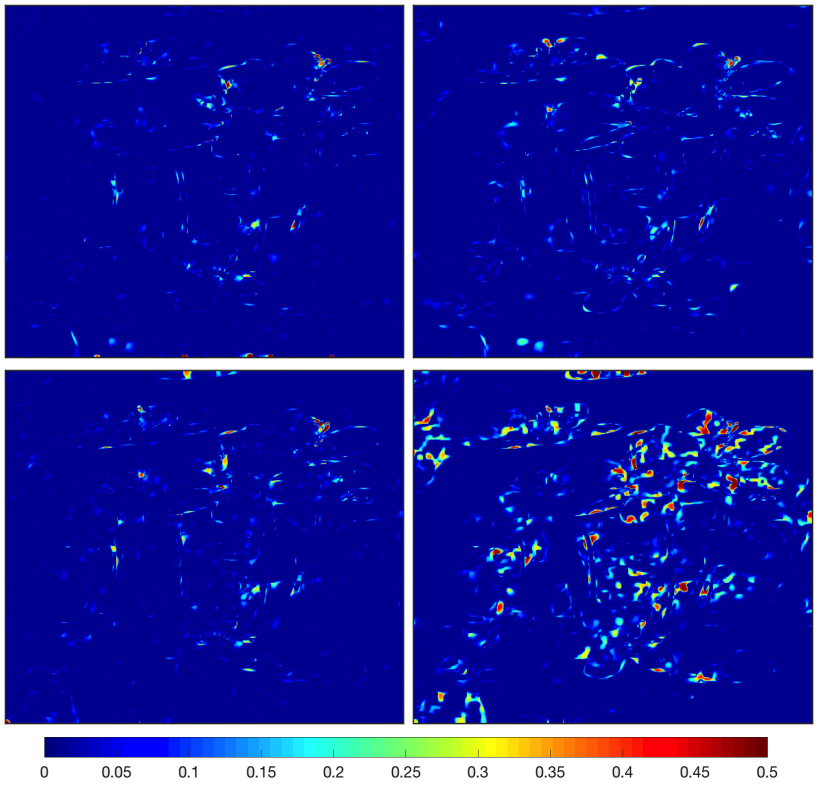


Figure 5.9: Difference from optimum when inpainting with different starting values. Left column: DG. Right column: iPiano. Top: Random initial value. Bottom: Unicolor initial value.

gradient. This rate does not depend on the problem size n, but rather on the dependency radius R of V. Numerical tests confirm the quality of images denoised and inpainted with Euler's elastica as a regularizer, that the time step adaptivity proposed in Algorithm 5.2 can improve execution time, and that our algorithm performs faster than the iPiano algorithm in certain instances.

There are still open questions, two of which carry special importance. Firstly, it should be possible to improve upon the time step adaptivity of Algorithm 5.2 which, while effective in some instances, is rudimentary. It may be possible to employ a stochastically ordered version as in [19] instead. Secondly, one should investigate the convergence properties of Algorithm 5.1 when applied to non-differentiable problems since it is still applicable then. One may also generalize Algorithm 5.1 to a manifold setting using the tools developed in [7]. Finally, one may wish to apply the discrete gradient approach to other non-convex optimization problems.

Figure 5.10: Inpainting with different starting values. Left column: DG. Right column: iPiano. Top: Random initial value. Middle: Unicolor (black) initial value. Bottom: Original initial value.

Acknowledgements

CBS acknowledges the support of the Engineering and Physical Sciences Research Council (EPSRC) 'EP/K009745/1', the EPSRC grant 'EP/M00483X/1', the EPSRC centre 'EP/N014588/1', the Leverhulme Trust project 'Breaking the non-convexity barrier', the Alan Turing Institute 'TU/B/000071', the Isaac Newton Institute, the Cantab Capital Institute for the Mathematics of Information and from NoMADS (Horizon 2020 RISE project grant). This work was supported by the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No. 691070. The authors wish to thank Antonin Chambolle, Elena Celledoni, Brynjulf Owren, and Reinout Quispel for their helpful discussions during this work. We also wish to thank the anonymous referees for their thorough advice.

Bibliography

[1] E. BAE, X.-C. TAI, AND W. ZHU, *Augmented Lagrangian method for an Euler's elastica based segmentation model that promotes convex contours*, Inverse Probl. Imag., 11 (2017), pp. 1–23.

[2] A. BECK AND L. TETRUASHVILI, *On the convergence of block coordinate descent type methods*, SIAM J. Optimiz., 23 (2013), pp. 2037–2060.

[3] D. P. BERTSEKAS, *Nonlinear programming*, Athena Scientific Optimization and Computation Series, Athena Scientific, Belmont, MA, second ed., 1999.

[4] J. BOLTE, S. SABACH, AND M. TEBOULLE, *Proximal alternating linearized minimization for nonconvex and nonsmooth problems*, Math. Program., 146 (2014), pp. 459–494.

[5] K. BREDIES, T. POCK, AND B. WIRTH, *A convex, lower semicontinuous approximation of Euler's elastica energy*, SIAM J. Math. Anal., 47 (2015), pp. 566–613.

[6] R. P. BRENT, *An algorithm with guaranteed convergence for finding a zero of a function*, Comput. J., 14 (1971), pp. 422–425.

[7] E. CELLEDONI AND B. OWREN, *Preserving first integrals with symmetric Lie group methods*, Discrete Cont. Dyn. S., 34 (2014), pp. 977–990.

[8] A. CHAMBOLLE AND T. POCK, *A first-order primal-dual algorithm for convex problems with applications to imaging*, J. Math. Imaging Vis., 40 (2011), pp. 120–145.

[9] A. CHAMBOLLE AND T. POCK, *Total roto-translational variation*, arXiv preprint arXiv:1709.09953, (2017).

[10] M. J. EHRHARDT, E. S. RIIS, T. RINGHOLM, AND C.-B. SCHÖNLIEB, *A geometric integration approach to smooth optimisation: Foundations of the discrete gradient method*, arXiv preprint arXiv:1805.06444, (2018).

[11] O. GONZALEZ, *Time integration and discrete Hamiltonian systems*, J. Nonlinear Sci., 6 (1996), pp. 449–467.

[12] V. GRIMM, R. I. MCLACHLAN, D. I. MCLAREN, G. QUISPEL, AND C. SCHÖNLIEB, *Discrete gradient methods for solving variational image regularisation models*, J. Phys. A: Math. Theor., 50 (2017), p. 295201.

[13] E. HAIRER AND C. LUBICH, *Energy-diminishing integration of gradient systems*, IMA J. Numer. Anal., 34 (2013), pp. 452–461.

[14] A. HARTEN, P. D. LAX, AND B. VAN LEER, *On upstream differencing and Godunov-type schemes for hyperbolic conservation laws*, SIAM Rev., 25 (1983), pp. 35–61.

[15] T. ITOH AND K. ABE, *Hamiltonian-conserving discrete canonical equations based on variational difference quotients*, J. Comput. Phys., 76 (1988), pp. 85–102.

[16] H. KARIMI, J. NUTINI, AND M. SCHMIDT, *Linear convergence of gradient and proximal-gradient methods under the Polyak–Łojasiewicz condition*, in ECML PKDD, Springer, 2016, pp. 795–811.

[17] R. I. MCLACHLAN, G. R. W. QUISPEL, AND N. ROBIDOUX, *Geometric integration using discrete gradients*, Philos. T. R. Soc. A, 357 (1999), pp. 1021–1045.

[18] Y. NESTEROV, *A method of solving a convex programming problem with convergence rate* $\mathcal{O}(1/k^2)$, Sov. Math. Dokl., 27 (1983), pp. 372–376.

[19] Y. NESTEROV, *Efficiency of coordinate descent methods on huge-scale optimization problems*, SIAM J. Optim, 22 (2012), pp. 341–362.

[20] Y. NESTEROV, *Gradient methods for minimizing composite functions*, Math. Program., 140 (2013), pp. 125–161.

[21] Y. NESTEROV AND A. NEMIROVSKII, *Interior-point polynomial algorithms in convex programming*, vol. 13, SIAM, 1994.

[22] M. NITZBERG, D. MUMFORD, AND T. SHIOTA, *Filtering, Segmentation and Depth*, vol. 662 of Lecture Notes in Comput.Sci., Springer, Berlin, 1993.

[23] P. OCHS, Y. CHEN, T. BROX, AND T. POCK, *iPiano: Inertial proximal algorithm for nonconvex optimization*, SIAM J. Imaging Sci., 7 (2014), pp. 1388–1419.

[24] B. T. POLYAK, *Some methods of speeding up the convergence of iteration methods*, USSR Comp. Math. Math+, 4 (1964), pp. 1–17.

[25] W. H. PRESS, *Numerical recipes 3rd edition: The art of scientific computing*, Cambridge university press, 2007.

[26] L. I. RUDIN, S. OSHER, AND E. FATEMI, *Nonlinear total variation based noise removal algorithms*, Physica D: Nonlinear Phenomena, 60 (1992), pp. 259–268.

[27] D. SCIEUR, V. ROULET, F. BACH, AND A. D'ASPREMONT, *Integration methods and optimization algorithms*, in Adv. Neur. In. 30, 2017, pp. 1109–1118.

[28] J. SHEN, H. SUNG, AND T. CHAN, *Euler's elastica and curvature-based inpainting*, SIAM J. Appl. Math., 63 (2003), pp. 564–592.

[29] N. Z. SHOR, *Minimization methods for non-differentiable functions*, Springer, 1985.

[30] X.-C. TAI, J. HAHN, AND G. J. CHUNG, *A fast algorithm for Euler's elastica model using augmented Lagrangian method*, SIAM J. Imaging Sci., 4 (2011), pp. 303–344.

[31] P. WOLFE, *Convergence conditions for ascent methods*, SIAM Rev., 11 (1969), pp. 226–235.

[32] S. J. WRIGHT, *Coordinate descent algorithms*, Math. Program., 151 (2015), pp. 3–34.

[33] D. XIE AND L. ADAMS, *New parallel SOR method by domain partitioning*, SIAM J. Sci. Comput., 20 (1999), pp. 2261–2281.

[34] J. ZHANG AND K. CHEN, *A new augmented Lagrangian primal dual algorithm for elastica regularization*, J. Alg. Comp. Tech., 10 (2016), pp. 325–338.

[35] J. ZHANG, R. CHEN, C. DENG, AND S. WANG, *Fast linearized augmented Lagrangian method for Euler's elastica model*, Numer. Math. - Theory Me., 10 (2017), pp. 98–115.

[36] W. ZHU, X.-C. TAI, AND T. CHAN, *Augmented Lagrangian method for a mean curvature based image denoising model*, Inverse Probl. Imag., 7 (2013), pp. 1409–1432.

A geometric integration approach to smooth optimisation: Foundations of the discrete gradient method

Matthias Ehrhardt, Erlend Riis, Torbjørn Ringholm and Carola-Bibiane Schönlieb

To be submitted

A geometric integration approach to smooth optimisation: Foundations of the discrete gradient method

Abstract. Discrete gradient methods are tools from geometric integration which yield optimisation schemes that inherit the energy dissipation property from continuous gradient flow. They are efficient for both convex and nonconvex problems, and by choosing different discrete gradients, one can obtain both zero- and first-order optimisation algorithms. In this paper, we make a comprehensive analysis of discrete gradient methods in optimization, answering questions about well-posedness of the iterates, convergence rates and optimal step size selection. In particular, we prove under mild assumptions that the iterates are well-posed, i.e. admit a solution, for all time steps, an unprecedented result for discrete gradients. We show that these schemes achieve $\mathcal{O}(1/k)$ and linear convergence rates, under standard assumptions on the objective function, such as smoothness, strong convexity or the Polyak-Łojasiewicz property. Furthermore, we recover the optimal rates of classical schemes such as explicit gradient descent and stochastic coordinate descent. The analysis is carried out for three discrete gradients: The Gonzalez discrete gradient, the mean value discrete gradient, the Itoh–Abe discrete gradient, and a randomised version of the Itoh–Abe method.

6.1 Introduction

Discrete gradients are tools from geometric integration for numerically solving first-order systems of ordinary differential equations (ODEs), while ensuring that certain structures of the continuous system—specifically energy conservation and dissipation, and Lyapunov functions—are preserved in the numerical solution. The use of discrete gradient methods to solve optimisation problems has gained increasing attention in recent years, because discrete gradients applied to the gradient flow ODE preserve energy dissipation for the objective function. This means that the iterative scheme monotonically decreases the objective function for all time steps, and the rate of dissipation is a discretised version of the rate of dissipation of gradient flow.

We consider the unconstrained optimisation problem

$$\min_{x\in\mathbb{R}^n} V(x), \tag{6.1.1}$$

where the function $V : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable. The discrete gradient method is of the form

$$x^{k+1} = x^k - \tau_k \overline{\nabla} V(x^k, x^{k+1}), \tag{6.1.2}$$

where $\tau_k > 0$ is the time step, and $\overline{\nabla} V$ is the discrete gradient, to be defined in Section 6.2.

6.1.1 Contributions

While discrete gradient methods have existed in geometric integration since the 1980s, only recently have they been studied in the context of optimisation, leaving significant gaps in our understanding of these schemes. In this paper, we resolve fundamental questions about the discrete gradient methods, including their well-posedness, efficiency, and optimal tuning. Specifically, we prove that the discrete gradient method's update formula (6.1.2) is well-posed, meaning that for any time step $\tau_k > 0$ and $x^k \in \mathbb{R}^n$, a solution x^{k+1} exists, under mild assumptions on V. We also analyse the dependence of the iterates on the choice of time step τ_k, which naturally lead to optimal time steps. Finally, we establish convergence rates for functions with Lipschitz continuous gradients, and functions that satisfy the Polyak–Łojasiewicz (PL) inequality [23]. We emphasise that the majority of these results hold for nonconvex functions. Our contributions to the foundations of the discrete gradient method opens the door for future applications and research on discrete gradient methods for optimisation, with a deeper understanding of their numerical properties.

6.1.2 Background and related work

We list some applications of discrete gradient methods for optimisation. In [16], discrete gradient methods are used to solve variational problems in image analysis. Furthermore, it is proven that the method converges to a set of stationary points. In [39], a discrete gradient method is applied to nonconvex imaging problems regularised with Euler's elastica. In [27], it was shown that the well-known Gauss-Seidel and successive-over-relaxation (SOR) methods are instances of the Itoh–Abe discrete gradient method for solving linear systems. In [13], the application of a derivative-free discrete gradient method for nonconvex, nonsmooth functions is studied, and the method converges to a set of stationary points in the Clarke subdifferential framework. In [10], an extension of the Itoh–Abe discrete gradient for the optimisation of functions defined on Riemannian manifolds is investigated. In [20], a discrete gradient method is combined with Hopfield networks in order to preserve a Lyapunov function for optimisation problems.

More generally, there is a wide range of research areas that study connections between optimisation schemes and systems of ODEs. We mention two prominent examples here. Recent papers [46, 47] study second-order differential equations that can be viewed as the continuous-time limit of Nesterov's accelerated gradient descent [29], in order to gain a deeper understanding of

the well-known acceleration method. Furthermore, in [43], the authors show that several accelerated optimisation schemes can be derived as multi-step integration schemes from numerical analysis. Another example is the study of gradient flows in metric spaces and minimising movement schemes [2], which studies gradient flow trajectories under other measures of distance, such as the Wasserstein metric [41].

6.1.3 Notation

Throughout the paper, we will denote by S^{n-1} the unit sphere $\{x \in \mathbb{R}^n : \|x\| = 1\}$. The diameter of a set $K \subset \mathbb{R}^n$ is defined as $\operatorname{diam}(K) := \sup_{x,y\in K} \|x-y\|$. The line segment between two points is defined as $[x, y] := \{\lambda x + (1-\lambda)y : \lambda \in [0,1]\}$.

In this paper, we consider both deterministic and stochastic schemes. For the stochastic schemes, there is a random distribution Ξ on S^{n-1} such that each iterate x^k depends on a descent direction d^k which is independently drawn from Ξ. We denote by ξ^k the joint distribution of $(d^i)_{i=1}^k$. We denote by ϕ_{k+1} the expectation of $V(x^{k+1})$ conditioned on ξ^k,

$$\phi_{k+1} := \mathbb{E}_{\xi^k}[V(x^{k+1})]. \tag{6.1.3}$$

To unify notation for all the methods in this paper, we will write ϕ_{k+1} instead of $V(x^{k+1})$ for the deterministic methods as well.

6.1.4 Structure

The rest of the paper is structured as follows. In Section 6.2 we define discrete gradients and introduce the four discrete gradient methods considered in this paper. In Section 6.3, the existence result for the discrete rgadient method (6.1.2) is given. In Section 6.4, we analyse the dependence of the iterates on the choice of time step, and obtain estimates for preferable time steps in the cases of L-smoothness and strong convexity. In Section 6.5, we prove convergence rates of the four methods for L-smooth functions, and for L-smooth PL functions—the latter meaning functions that satisfy the PL inequality. In Section 6.6, we briefly discuss a preconditioned extension of the discrete gradient method. In Section 6.7, we conclude and present an outlook for future work.

6.2 Discrete gradient methods

6.2.1 Discrete gradients and gradient flow

Consider the gradient flow of V,

$$\dot{x} = -\nabla V(x), \qquad x(0) = x_0 \in \mathbb{R}^n, \tag{6.2.1}$$

where the dot is the derivative of x with respect to time. This system is fundamental to optimisation, and underpins many gradient-based iterative schemes. By applying the chain rule, we obtain

$$\frac{\mathrm{d}}{\mathrm{d}t} V(x(t)) = \langle \nabla V(x(t)), \dot{x}(t) \rangle = -\|\nabla V(x(t))\|^2 = -\|\dot{x}(t)\|^2 \leq 0. \tag{6.2.2}$$

Hence, gradient flow has an energy dissipative structure, since the function value $V(x(t))$ decreases monotonically along any solution $x(t)$ to (6.2.1). Furthermore, the rate of dissipation is given in terms of the norm of both ∇V and $\dot{x}$.

In geometric integration, one studies methods for solving ODEs numerically while preserving certain structures of the continuous system—see [18,24] for an introduction. Discrete gradients are tools for solving first-order ODEs that preserve energy conservation laws, dissipation laws, and Lyapunov functions [15,22,25,38]. They are defined as follows.

Definition 6.1 (Discrete gradient)**.** Let V be a continuously differentiable function. A *discrete gradient* is a continuous map $\overline{\nabla} V : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$ such that for all $x, y \in \mathbb{R}^n$, we have

$$\langle \overline{\nabla} V(x, y), y - x \rangle = V(y) - V(x) \qquad \text{(Mean value property)}, \tag{6.2.3}$$

$$\lim_{y \to x} \overline{\nabla} V(x, y) = \nabla V(x) \qquad \text{(Consistency property)}. \tag{6.2.4}$$

For a sequence of strictly positive time steps $(\tau_k)_{k \in \mathbb{N}}$ and a starting point $x^0 \in \mathbb{R}^n$, the discrete gradient method applied to (6.2.1) is given by (6.1.2), i.e.

$$x^{k+1} = x^k - \tau_k \overline{\nabla} V(x^k, x^{k+1}).$$

This scheme preserves the dissipative structure of gradient flow, as can be seen by applying the mean value property of discrete gradients.

$$\begin{aligned} V(x^{k+1}) - V(x^k) &= \langle \overline{\nabla} V(x^k, x^{k+1}), x^{k+1} - x^k \rangle \\ &= -\tau_k \|\overline{\nabla} V(x^k, x^{k+1})\|^2 \qquad (6.2.5) \\ &= -\frac{1}{\tau_k} \|x^{k+1} - x^k\|^2. \qquad (6.2.6) \end{aligned}$$

Similarly to the dissipation law (6.2.2) of gradient flow, the decrease of the objective function value is given in terms of the norm of both the step $x^{k+1} - x^k$ and of the discrete gradient. Furthermore, the decrease holds for arbitrary τ_k, so the method imposes no restraint on the time steps.

Throughout the paper, we assume that there are bounds $\tau_{\max} \geq \tau_{\min} > 0$ such that for all $k \in \mathbb{N}$,

$$\tau_{\min} \leq \tau_k \leq \tau_{\max}.$$

However, no restrictions are required on either of these bounds. In [16], it is proven that if V is continuously differentiable and coercive—the latter meaning that the level set $\{x \in \mathbb{R}^n : V(x) \leq M\}$ is bounded for each $M \in \mathbb{R}$—and if the time steps τ_k are between $\tau_{\min}$ and $\tau_{\max}$, then the iterates $(x^k)_{k\in\mathbb{N}}$ of (6.1.2) converge to a set of stationary points, i.e. points $x^* \in \mathbb{R}^n$ such that

$$\nabla V(x^*) = 0.$$

We may compare the discrete gradient method with explicit gradient descent,

$$x^{k+1} = x^k - \tau_k \nabla V(x^k).$$

Unlike discrete gradient methods, gradient descent is only guaranteed to decrease the objective function value for sufficiently small time steps τ_k. To ensure decrease and convergence for this scheme, the time steps must be restricted based on estimates of the smoothness of the gradient of V, which might not be available, or lead to prohibitively small time steps. We mention that one may also consider other numerical integration methods, such as implicit Runge-Kutta methods, where energy dissipation is ensured under mild time step restrictions [17].

6.2.2 Four discrete gradient methods

We now introduce the four discrete gradient methods considered in this paper. The first three methods use the well-known the Gonzalez discrete gradient, mean value discrete gradient, and the Itoh–Abe discrete gradient. The fourth method is a randomised generalisation of the Itoh–Abe method, from hereon referred to as the randomised Itoh–Abe method.

1. *The Gonzalez discrete gradient* [15] (also known as the midpoint discrete gradient) is given by

$$\overline{\nabla} V(x, y) = \nabla V\left(\frac{x+y}{2}\right) + \frac{V(y) - V(x) - \langle \nabla V(\frac{x+y}{2}), y - x\rangle}{\|x - y\|^2}(y - x), \quad x \neq y.$$

2. *The mean value discrete gradient* [19], used for example in the average vector field method [11], is given by

$$\overline{\nabla} V(x, y) = \int_0^1 \nabla V\big((1-s)x + sy\big)\,\mathrm{d}s.$$

3. *The Itoh–Abe discrete gradient* [22] (also known as the coordinate incre-

ment discrete gradient) is given by

$$\overline{\nabla} V(x,y) = \begin{pmatrix} \frac{V(y_1,x_2,\ldots,x_n)-V(x)}{y_1-x_1} \\ \frac{V(y_1,y_2,x_3,\ldots,x_n)-V(y_1,x_2,\ldots,x_n)}{y_2-x_2} \\ \vdots \\ \frac{V(y)-V(y_1,\ldots,y_{n-1},x_n)}{y_n-x_n} \end{pmatrix},$$

where $0/0$ is interpreted as $\partial_i V(x)$.

While the first two discrete gradients are gradient-based and can be seen as approximations to the midpoint gradient $V\left(\frac{x+y}{2}\right)$, the Itoh–Abe discrete gradient is derivative-free, and is evaluated by computing successive, coordinate-wise difference quotients. In an optimisation setting, the Itoh–Abe discrete gradient is often preferable to the others, as it is relatively computationally inexpensive. Solving the implicit equation (6.1.2) with this discrete gradient amounts to successively solving n scalar equations of the form

$$\begin{aligned} x_1^{k+1} &= x_1^k - \tau_k \frac{V(x_1^{k+1}, x_2^k, \ldots, x_n^k) - V(x^k)}{x_1^{k+1} - x_1^k} \\ x_2^{k+1} &= x_2^k - \tau_k \frac{V(x_1^{k+1}, x_2^{k+1}, x_3^k, \ldots, x_n^k) - V(x_1^{k+1}, x_2^k, \ldots, x_n^k)}{x_2^{k+1} - x_2^k} \\ &\vdots \\ x_n^{k+1} &= x_n^k - \tau_k \frac{V(x^{k+1}) - V(x_1^{k+1}, x_2^{k+1}, \ldots, x_n^{k+1}, x_n^k)}{x_n^{k+1} - x_n^k}. \end{aligned}$$

4. *The Randomised Itoh–Abe method* [13] is an extension of the Itoh–Abe discrete gradient method, wherein the directions of descent are randomly chosen. We consider a sequence of independent, identically distributed directions $(d^k)_{k\in\mathbb{N}} \subset S^{n-1}$ drawn from a random distribution Ξ, and solve

$$x^{k+1} = x^k - \tau_k \frac{V(x^{k+1}) - V(x^k)}{\langle x^{k+1} - x^k, d^{k+1}\rangle} d^{k+1},$$

This can be rewritten as solving

$$x^{k+1} \mapsto x^k - \tau_k \alpha_k d^{k+1}, \quad \alpha_k = -\frac{V(x^k - \tau_k \alpha_k d^{k+1}) - V(x^k)}{\tau_k \alpha_k},$$

where $x^{k+1} = x^k$ is considered a solution whenever $\langle \nabla V(x^k), d^{k+1}\rangle = 0$.

We also define the constant

$$\zeta := \min_{e\in S^{n-1}} \mathbb{E}_{d\sim\Xi}[\langle d, e\rangle^2], \tag{6.2.7}$$

and assume that Ξ is such that $\zeta > 0$. For example, both for the uniform random distribution on S^{n-1} and the uniform random distribution on the standard coordinates $(e^i)_{i=1}^n$, we have

$$\zeta = \frac{1}{n}.$$

See [44, Table 4.1] for estimates of (6.2.7) for the above cases and others.

This scheme is a generalisation of the Itoh–Abe discrete gradient method, in that the methods are equivalent if $(d^k)_{k\in\mathbb{N}}$ cycle through the standard coordinates with the rule

$$d^k = e^{[(k-1) \bmod n]+1}, \quad k = 1, 2, \ldots$$

However, the computational effort of one iterate of the Itoh–Abe discrete gradient method is equal to n steps of the randomised method, so the efficiency of the methods should be judged accordingly.

While this method does not retain the discrete gradient structure of the Itoh–Abe discrete gradient, the dissipativity properties can be rewritten in the following manner.

$$V(x^{k+1}) - V(x^k) = -\tau_k \left(\frac{V(x^{k+1}) - V(x^k)}{\|x^{k+1} - x^k\|} \right)^2, \tag{6.2.8}$$

$$V(x^{k+1}) - V(x^k) = -\frac{1}{\tau_k} \|x^{k+1} - x^k\|^2. \tag{6.2.9}$$

The motivation for introducing this randomised extension of the Itoh–Abe method is, first, to tie in discrete gradient methods with other optimisation methods such as stochastic coordinate descent [14, 37, 48] and random pursuit [32,44], and, second, because this method extends to the nonsmooth, nonconvex setting [13].

6.3 Existence of solution of discrete gradient step

In this section, we prove that the discrete gradient equation

$$y = x - \tau \overline{\nabla} V(x, y). \tag{6.3.1}$$

admits a solution y, for all time steps $\tau > 0$ and points $x \in \mathbb{R}^n$, under mild assumptions on V and $\overline{\nabla} V$. The result applies to the three discrete gradients considered in this paper, and we expect that it also covers a vast number of other discrete gradients. We also prove that the randomised Itoh–Abe method admits a solution for all τ and x in a subsequent proposition. We note that these results do not require convexity of V.

In [35], an existence and uniqueness result for sufficiently small time steps is given for a large class of discrete gradients. This uses the Banach fixed point theorem, which also allows them to approximate the solution by a contraction mapping. Furthermore, the existence of a solution for the Gonzalez discrete gradient is established for sufficiently small time steps via the implicit function theorem in [45, Theorem 8.5.4]. To the authors' knowledge, the following result is the first without a restriction on time steps.[1]

We use the following notation. For $\delta > 0$, the closed ball of radius δ about x is defined as $\overline{B}_\delta(x) := \{y \in \mathbb{R}^n : \|y - x\| \le \delta\}$. For a set $K \subset \mathbb{R}^n$, we define the δ-thickening, $K_\delta = \{x \in \mathbb{R}^n : \operatorname{dist}(K, x) \le \delta\}$. The convex hull of K is denoted by $\operatorname{co} K$.

We make the following assumption for the discrete gradient.

Assumption 6.1. *There is a constant C_n that depends on the discrete gradient but is independent of V, and a continuous, nondecreasing function $\delta : [0,\infty] \to [0,\infty]$, where $\delta(0) = 0$ and $\delta(\infty) := \lim_{r\to\infty} \delta(r)$, such that the following holds.*

For any $V \in C^1(\mathbb{R}^n;\mathbb{R})$ and any convex set $K \subset \mathbb{R}^n$ with nonempty interior, the two following properties are satisfied.

(i) If $\|\nabla V(x)\| \le L$ for all $x \in K_{\delta(\operatorname{diam}(K))}$, then $\|\overline{\nabla} V(x,y)\| \le C_n L$ for all $x, y \in K$.

(ii) If W is another continuously differentiable function such that $V(x) = W(x)$ for all $x \in K_{\delta(\operatorname{diam}(K))}$, then $\overline{\nabla} V(x,y) = \overline{\nabla} W(x,y)$ for all $x, y \in K$.

The following result, which is proved in Appendix 6.A, shows that the discrete gradients considered in this paper satisfy the above assumption.

Lemma 6.1. *The three discrete gradients satisfy Assumption 6.1 with the following constants.*

1. *For the Gonzalez discrete gradient,*
$$C_n = \sqrt{2}, \qquad \delta \equiv 0.$$
2. *For the mean value discrete gradient,*
$$C_n = 1, \qquad \delta \equiv 0.$$
3. *For the Itoh–Abe discrete gradient,*
$$C_n = \sqrt{n}, \qquad \delta(r) = r.$$

[1] Note that previous existence results were derived in the context of numerical integration, where small time steps are necessary, in order to solve the ODE as accurately as possible. On the other hand, for optimisation, we mainly care about the preservation of dissipation, which holds for all time steps.

Remark 6.1. *We note that for the Gonzalez and mean value discrete gradients, the bounds C_n are independent of the dimension, and the existence analysis could therefore be extended to infinite dimensions, by invoking Schauder's fixed point theorem [42] and replacing continuity with weak continuity.*

We state the well-known Brouwer fixed point theorem [7], a key argument for many existence theorems.

Proposition 6.1 (Brouwer fixed point theorem)**.** *Let $K \subset \mathbb{R}^n$ be a convex, compact set and $g : K \to K$ a continuous function. Then g has a fixed point in K.*

We proceed to state and prove the existence theorem.

Theorem 6.1 (Discrete gradient existence theorem)**.** *Suppose V is continously differentiable and that $\overline{\nabla}$ satisfies Assumption 6.1. Then there exists a solution y to* (6.3.1) *for any $\tau > 0$ and $x \in \mathbb{R}^n$, if V satisfies either of the following properties.*

(i) The gradient of V is uniformly bounded.

(ii) V is coercive.

(iii) The function δ in Assumption 6.1 is equal to 0, and both V and the gradient of V are uniformly bounded on the convex hull of the level set $\{y : V(y) \leq V(x)\}$ (the bounds may depend on x).

Proof. (i). We define the function

$$g(y) = x - \tau \overline{\nabla} V(x, y),$$

and want to show that it has a fixed point, $y = g(y)$. There is $L > 0$ such that $\|\nabla V(y)\| \leq L$ for all $y \in \mathbb{R}^n$. Therefore, by Assumption 6.1,

$$\|\overline{\nabla} V(x, y)\| \leq C_n L$$

for all $y \in \mathbb{R}^n$. This implies that $g(y) \in \overline{B}_{\tau C_n L}(x)$ for all $y \in \mathbb{R}^n$. Specifically, g maps $\overline{B}_{\tau C_n L}(x)$ into itself. As g is continuous, it follows from Brouwer's fixed point theorem that there exists a point $y \in \overline{B}_{\tau C_n L}(x)$ such that $g(y) = y$, and we are done.

(ii). Let $\sigma > 0$, let K be the convex hull of the level set

$$K = \operatorname{co}\Big(\{y : V(y) \leq V(x)\}\Big)$$

and set $\delta = \delta(\mathrm{diam}(K))$. Since V is coercive, K_δ and $K_{\delta+\sigma}$ are bounded. By standard arguments [33, Corollary 2.5], there exists a cutoff function $\varphi \in \mathcal{C}_c^\infty(\mathbb{R}^n; [0,1])$ such that

$$\varphi(y) = \begin{cases} 1 & \text{if } y \in K_\delta, \\ 0 & \text{if } y \notin K_{\delta+\sigma}. \end{cases}$$

We define $W : \mathbb{R}^n \to \mathbb{R}$ by

$$W(y) := \varphi(y)\big(V(y) - V(x)\big) + V(x).$$

W is continuously differentiable and $\operatorname{supp} \nabla W \subset K_{\delta+\sigma}$. Therefore, W has uniformly bounded gradient, so by part *(i)* there is a y such that

$$y = x - \tau \overline{\nabla} W(x, y).$$

By (6.2.5), $W(y) < W(x)$ which implies that $y \in K_\delta$, so $W(y) = V(y)$. Furthermore, since $W(x) = V(x)$, we deduce that $V(y) < V(x)$, so $y \in K$. Lastly, since V and W coincide on K_δ, and x and y both belong to K, it follows from Assumption 6.1, property *(ii)* that $\overline{\nabla} V(x, y) = \overline{\nabla} W(x, y)$. Hence a solution $y = x - \tau \overline{\nabla} V(x, y)$ exists.

(iii). Set $K = \mathrm{co}\Big(\big\{y : V(y) \le V(x)\big\}\Big)$, $L = \sup\big\{\|\nabla V(y)\| : V(y) \le M + \varepsilon\big\}$, and $M = \sup_{y \in K} V(y)$. Furthermore let $\varepsilon > 0$ and set $F = \big\{y : V(y) \ge M + \varepsilon\big\}$. The mean value theorem [34, Equation A.55] and the boundedness of ∇V imply that for all $y \in K$ and $z \in F$, there is $\lambda \in (0,1)$ such that

$$\varepsilon \le |V(y) - V(z)| = |\langle \nabla V(\lambda y + (1-\lambda)z), y - z\rangle| \le L\|y - z\|.$$

Therefore, for all $y \in K$ and $z \in F$,

$$\|y - z\| \ge \frac{\varepsilon}{L}.$$

By Lemma 6.6, there exists a cutoff function $\varphi \in C^\infty(\mathbb{R}^n; [0,1])$ with uniformly bounded gradient, such that

$$\varphi(y) = \begin{cases} 1 & \text{if } y \in K, \\ 0 & \text{if } y \in F. \end{cases}$$

Consider $W : \mathbb{R}^n \to \mathbb{R}$ defined by

$$W(y) = \varphi(y)\big(V(y) - V(x)\big) + V(x).$$

The gradient of W is uniformly bounded, so there is a fixed point y such that

$$y = x - \tau \overline{\nabla} W(x, y).$$

By the same arguments as in case *(ii)*, $\overline{\nabla} V(x, y) = \overline{\nabla} W(x, y)$, which implies that y solves $y = x - \tau \overline{\nabla} V(x, y)$. □

The third case in the above theorem covers a wide range of optimisation problems for which the objective function does not necesssarily have bounded level sets. This covers important optimisation problems, such as the least squares optimisation of linear systems,

$$V(x) = \frac{1}{2}\|Ax - f\|^2,$$

where the kernel of A is nonempty.

While the above theorem holds also for the Itoh–Abe discrete gradient, there is a much simpler existence result, which covers the randomised Itoh–Abe method and, by extension, the Itoh–Abe discrete gradient method, and which only requires the objective function to be bounded below—in fact, it only requires continuity, not differentiability, of V [13]. For completeness, we present a short proof of this, in the differentiable setting.

Proposition 6.2. *Suppose V is bounded below. Then, for all $\tau > 0$ and $x \in \mathbb{R}^n$, there is an $\alpha \in \mathbb{R}$ such that*

$$y = x - \tau\alpha d, \qquad \text{where } \alpha \neq 0 \text{ solves} \qquad -\alpha = \frac{V(x - \tau\alpha d) - V(x)}{\tau\alpha},$$

where $\alpha = 0$ is considered a solution if $\langle \nabla V(x), d\rangle = 0$.

Proof. Suppose $\langle \nabla V(x), d\rangle \neq 0$, and, without loss of generality, choose $\varepsilon > 0$ such that $\langle \nabla V(x), d\rangle \geq \varepsilon$. Then for sufficiently small $\alpha_1 > 0$, we have

$$\frac{V(x - \tau\alpha_1 d) - V(x)}{\tau\alpha_1} \leq -\frac{\varepsilon}{2} \leq -\alpha_1.$$

On the other hand, since V is bounded below, then either there is $\alpha > 0$ such that $V(x - \tau\alpha d) = V(x)$, or

$$\frac{V(x - \tau\alpha d) - V(x)}{\tau\alpha} \to 0 \quad \text{as } \alpha \to +\infty.$$

In either case, there is $\alpha_2 > 0$ sufficiently large so that

$$\frac{V(x - \tau\alpha_2 d) - V(x)}{\tau\alpha_2} \geq -\alpha_2.$$

By the intermediate value theorem [40, Theorem 4.23], there exists $\alpha > 0$ such that

$$-\alpha = \frac{V(x - \tau\alpha d) - V(x)}{\tau\alpha},$$

which yields the desired result. □

6.4 Analysis of time steps

In this section, we study the dependence of $x^k \mapsto x^{k+1}$ on the choice of time step τ for the randomised Itoh–Abe method. For both L-smooth, convex functions and strongly convex functions, we ascertain bounds on optimal time steps with respect to the decrease in V.

We fix a starting point x, direction $d \in S^{n-1}$ and time step τ, and study the solution y to

$$y = x - \tau\alpha d, \qquad \text{where } \alpha \neq 0 \text{ solves} \qquad -\alpha = \frac{V(x - \tau\alpha d) - V(x)}{\tau\alpha}. \tag{6.4.1}$$

By Proposition 6.2, a solution for y always exists. For convenience and to avoid the case $y = x$, we assume $\langle \nabla V(x), d\rangle > 0$. For notational brevity, we define the scalar function $f : \mathbb{R} \to \mathbb{R}$ as

$$f(\alpha) = V(x - \alpha d) - V(x).$$

Solving (6.4.1) is equivalent to solving

$$\frac{f(\alpha)}{\alpha^2} = -\frac{1}{\tau}. \tag{6.4.2}$$

6.4.1 Implicit dependence on the time step for Itoh–Abe methods

For optimisation schemes with a time step τ, it is common to assume that the distance between x and y increases with the time step. For explicit schemes, this is naturally the case. However, in implicit schemes, such as the discrete gradient method, this is not always the case. We demonstrate this with a simple example in one dimension.

Example 6.1. *Define $V(x) := -x^3$ and $x = 0$. For $\tau > 0$,* (6.4.1) *is solved by*

$$y = \frac{1}{\tau}.$$

Then, as $\tau \to 0$, we have $y \to \infty$, and as $\tau \to \infty$, we have $y \to x$.

The above example illustrates that for nonconvex functions, decreasing the time step might lead to a larger step $x \mapsto y$ and vice versa. We now show that for convex functions, the distance $\|y - x\|$ does increase monotonically with τ.

Since $\langle \nabla V(x), d\rangle > 0$ by assumption, there is $r > 0$ such that $V(x - \alpha d) < V(x)$ for $\alpha \in (0, r)$. Set

$$R = \sup\left\{r \,:\, V(x - \alpha d) < V(x) \text{ for all } \alpha \in (0, r)\right\}.$$

Proposition 6.3. *Let V be convex. Then there is a continuous, strictly increasing bijection $\tau \mapsto \alpha(\tau)$ from $(0,\infty)$ to $(0,R)$, such that $\alpha(\tau)$ solves (6.4.2) for the time step τ.*

Proof. By Proposition 6.2, there is a corresponding solution $\alpha(\tau) > 0$ for all $\tau > 0$. We will show that if α_1 and α_2 solve (6.1.2) for τ_1 and τ_2 respectively, then $\alpha_2 > \alpha_1$ if and only if $\tau_2 > \tau_1$. To do so, we use the alternative characterisation of convex functions in one dimension, which states that

$$\alpha \mapsto \frac{f(\alpha) - f(0)}{\alpha} = \frac{f(\alpha)}{\alpha}$$

is monotonically nondecreasing in α. If $\alpha_2 > \alpha_1$, then this implies

$$\begin{aligned} \frac{f(\alpha_1)}{\alpha_1} &\leq \frac{f(\alpha_2)}{\alpha_2} < 0 \\ \frac{f(\alpha_1)}{\alpha_1^2} &< \frac{f(\alpha_2)}{\alpha_2^2} < 0, \end{aligned} \tag{6.4.3}$$

where the second inequality follows from the first inquality and that $\alpha_2 > \alpha_1$. By (6.4.2), we have

$$\frac{f(\alpha_1)}{\alpha_1^2} = -\frac{1}{\tau_1}, \qquad \frac{f(\alpha_2)}{\alpha_2^2} = -\frac{1}{\tau_2}.$$

Combining this with (6.4.3), we derive that $\tau_2 > \tau_1$. Thus α strictly increases with τ. Furthermore, by letting $\tau_2 \downarrow \tau_1$, we see from these equations that $\alpha_2 \downarrow \alpha_1$. Therefore, the dependence of α on τ is continuous.

Next, we show that $\alpha(\tau) \to 0$ as $\tau \to 0$. This can be seen by inspecting

$$\frac{f(\alpha(\tau))}{\alpha(\tau)} = -\frac{\alpha(\tau)}{\tau}.$$

The left-hand side is bounded by the derivative $f'(0) = -\langle \nabla V(x), d \rangle$. Hence, as τ goes to zero, $\alpha(\tau)$ must also go to zero to prevent the right-hand side from blowing up.

Last, we show that $\alpha(\tau) \to R$ as $\tau \to \infty$. By inspecting

$$\frac{f(\alpha(\tau))}{\alpha(\tau)^2} = -\frac{1}{\tau},$$

we see that as $\tau \to \infty$, the right-hand side goes to zero, so either $f(\alpha(\tau)) \to 0$ or $\alpha(\tau)^2 \to \infty$. There are two cases to consider, $R < \infty$ and $R = \infty$. If $R < \infty$, then $f(R) = 0$, which implies that $\alpha(\tau) \to R$. If $R = \infty$, then $f(\alpha) < -\varepsilon$ for some $\varepsilon > 0$ and for all $\alpha > 0$, from which it follows that $\alpha(\tau)^2 \to \infty = R$. This concludes the proof. □

Remark 6.2. *The above proposition can also be shown to hold for nondifferentiable, convex functions, by replacing the derivative with a subgradient.*

6.4.2 Lipschitz continuous gradients

The remaining part of this section is devoted to deriving bounds on optimal time steps, when the objective function has Lipschitz continuous gradients, or is strongly convex. We start with the case of Lipschitz continuous gradients, also referred to as *L-smoothness*.

Definition 6.2 (L-smooth)**.** A function $V : \mathbb{R}^n \to \mathbb{R}$ is *L-smooth* if its gradient is Lipschitz continuous with Lipschitz constant L, i.e. if for all $x, y \in \mathbb{R}^n$,

$$\|\nabla V(x) - \nabla V(y)\| \leq L\|x - y\|.$$

We state some basic properties of L-smooth functions.

Proposition 6.4. *If V is L-smooth, then the following properties also hold.*

(i) $V(y) - V(x) \leq \langle \nabla V(x), y - x\rangle + \frac{L}{2}\|y - x\|^2$ *for all* $x, y \in \mathbb{R}^n$.

(ii) $V(\lambda x + (1-\lambda)y) \geq \lambda V(x) + (1-\lambda)V(y) - \frac{\lambda(1-\lambda)L}{2}\|x - y\|^2$ *for all* $\lambda \in [0,1]$, $x, y \in \mathbb{R}^n$.

Proof. Property *(i)*. [6, Proposition A.24].

Property *(ii)*. It follows from property *(i)* that the function

$$x \mapsto \frac{L}{2}\|x\|^2 - V(x)$$

is convex, which in turn yields the desired inequality. □

We now state and prove the first result for bounds on optimal time steps, which states that any time step $\tau < 2/L$ is suboptimal. Recall the scalar function $f(\alpha) = V(x - \alpha d) - V(x)$.

Lemma 6.2. *Let V be convex and L-smooth, and denote by α_* the solution to* (6.4.2) *for $\tau_* = 2/L$. If α solves* (6.4.2) *for $\tau < 2/L$, then $f(\alpha) > f(\alpha_*)$.*

Proof. Let $\lambda \in (\tau L/2, 1)$, and plug in 0 and α/λ for y and x respectively in Proposition 6.4 *(ii)* to get, after rearranging,

$$\lambda f(\alpha/\lambda) \leq f(\alpha) + \frac{(1-\lambda)L}{2\lambda}\alpha^2.$$

Plugging in (6.4.2), we get

$$\lambda f(\alpha/\lambda) \leq \left(1 - \frac{(1-\lambda)\tau L}{2\lambda}\right) f(\alpha)$$

or equivalently

$$f(\alpha/\lambda) \leq \left(\frac{1}{\lambda} - \frac{(1-\lambda)\tau L}{2\lambda^2}\right) f(\alpha).$$

We want to show that $f(\alpha/\lambda) < f(\alpha)$, i.e. that

$$\frac{1}{\lambda} - \frac{(1-\lambda)\tau L}{2\lambda^2} > 1.$$

By rearranging and solving the quadratic expression, we see that this holds whenever $\lambda \in (\tau L/2, 1)$. Thus $f(\alpha/\lambda) < f(\alpha)$.

We recall from Proposition 6.3 that since f is convex, there is a continuous, increasing bijection $\tau \mapsto \alpha(\tau)$ such that $f(\alpha(\tau)) < 0$. Since $\alpha/\lambda > \alpha$ and $f(\alpha/\lambda) < f(\alpha)$, there is $\tau' > \tau$ such that α/λ solves (6.4.2) for τ'. In other words, whenever $\tau < 2/L$, there is $\tau' > \tau$ such that $f(\alpha(\tau')) < f(\alpha)$. By convexity of f, the desired result follows. □

6.4.3 Strong convexity

We now move from L-smoothness to strong convexity, for which we use the terminology *μ-convex*.

Definition 6.3 (Strong convexity)**.** A convex function $V : \mathbb{R}^n \to \mathbb{R}$ is *strongly convex with parameter μ*, also termed *μ-convex*, if either of the following (equivalent) conditions hold.

(i) The function $V(\cdot) - \frac{\mu}{2}\|\cdot\|^2$ is convex.

(ii) $V\big(\lambda x + (1-\lambda)y\big) \leq \lambda V(x) + (1-\lambda)V(y) - \frac{\mu}{2}\lambda(1-\lambda)\|x-y\|^2$ for all $x, y \in \mathbb{R}^n$, $\lambda \in [0,1]$.

We now give the second result for bounds on optimal time steps, which states that for strongly convex functions, any time step $\tau > 2/\mu$ yields a suboptimal decrease.

Lemma 6.3. *Let $V : \mathbb{R}^n \to \mathbb{R}$ be μ-convex and denote by α_* the solution to* (6.4.2) *for $\tau_* = 2/\mu$. If α solves* (6.4.2) *for $\tau > 2/\mu$, then $f(\alpha) > f(\alpha_*)$.*

Proof. Fix $\lambda \in (2/(\tau\mu), 1)$, and plug in 0 and α for y and x respectively in Definition 6.3 *(ii)* to get, after rearranging,

$$f(\lambda\alpha) \leq \lambda f(\alpha) - \frac{\mu\lambda(1-\lambda)}{2}\alpha^2.$$

Plugging in (6.4.2), we get

$$f(\lambda\alpha) \leq \left(\lambda + \frac{\tau\mu\lambda(1-\lambda)}{2}\right) f(\alpha).$$

We want to show that $f(\lambda\alpha) < f(\alpha)$, i.e. that

$$\lambda + \frac{\tau\mu\lambda(1-\lambda)}{2} > 1.$$

By rearranging and solving the quadratic expression, we find that this is satisfied if $\lambda \in (2/(\tau\mu), 1)$. Thus $f(\lambda\alpha) < f(\alpha)$. By arguing as in the proof to Lemma 6.2, we deduce that $f(\alpha_*) < f(\alpha)$, which concludes the proof. □

Remark 6.3. *This result also holds for strongly convex, nondifferentiable functions.*

6.5 Convergence rate analysis

In the previous section, we developed a formal intuition for the dependence of the step $x^k \mapsto x^{k+1}$ on the time step τ, and derived suitable time steps τ whenever we have estimates on strong convexity of the objective function or Lipschitz continuity of its gradients. In this section we derive convergence rates for these cases, and more generally, for functions that satisfy the Polyak–Łojasiewicz inequality. We follow the arguments in [3, 31], on convergence rates of coordinate descent.

We recall the notation in (6.1.3),

$$\phi_{k+1} := \mathbb{E}_{\xi^k} V(x^{k+1}),$$

where $\phi_{k+1} = V(x^{k+1})$ for deterministic methods. Central to the proofs of convergence rates will be an estimate of the form

$$\beta\Big(V(x^k) - \phi_{k+1}\Big) \geq \|\nabla V(x^k)\|^2,$$

from which a variety of useful results follow. We therefore begin by deriving this estimate for each of the four methods. We assume throughout that the time steps $(\tau_k)_{k\in\mathbb{N}}$ are bounded above and below by constants,

$$0 < \tau_{\min} \leq \tau_k \leq \tau_{\max}, \quad \text{for all } k \in \mathbb{N}.$$

We do not require convexity of V for these estimates.

In order to optimise the estimates of β for the Itoh–Abe discrete gradient method and the randomised Itoh–Abe method, we consider coordinate-wise Lipschitz constants for the gradient of V as well as a directional Lipschitz constant. For $i = 1, \dots, n$, we suppose $\partial_i V : \mathbb{R}^n \to \mathbb{R}^n$ is Lipschitz continuous with Lipschitz constant $L_i \leq L$. That is, for all $x, y \in \mathbb{R}^n$, we have

$$\|\partial_i V(x) - \partial_i V(y)\| \leq L_i \|x - y\|.$$

We denote by L_{sum} the ℓ^2-norm of the coordinate-wise Lipschitz constants,

$$L_{\text{sum}} = \left(\sum_{i=1}^{n} L_i^2\right)^{1/2} \in [L, \sqrt{n}L].$$

Furthermore, for a direction $d \in S^{n-1}$, we consider the Lipschitz continuity constant $\tilde{L}_d \leq L$, such that for all $x \in \mathbb{R}^n$ and $\alpha \in \mathbb{R}$, we get

$$|\langle \nabla V(x + \alpha d), d\rangle - \langle \nabla V(x), d\rangle| \leq \tilde{L}_d |\alpha|.$$

We define $\tilde{L}_{\max} \leq L$ to be the supremum of $\tilde{L}_d$ over all d in the support of the probability density function of Ξ. That is, for all $d \sim \Xi$, all $x \in \mathbb{R}^n$, and all $\alpha \in \mathbb{R}$, it holds that

$$|\langle \nabla V(x + \alpha d), d\rangle - \langle \nabla V(x), d\rangle| \leq \tilde{L}_{\max} |\alpha|.$$

In this setting, we can refine the L-smoothness property in Proposition 6.4 *(i)* to

$$V(x + \alpha d) - V(x) \leq \alpha \langle \nabla V(x), d\rangle + \frac{\tilde{L}_{\max}}{2} \alpha^2, \tag{6.5.1}$$

for all $\alpha \in \mathbb{R}$ and d in the support of the density of Ξ [3, Lemma 3.2].

When Ξ only draws from the standard coordinates $(e^i)_{i=1}^n$, for many large-scale optimisation problems, $\tilde{L}_{\max}$ will be significantly smaller than L [48]. In other cases, where the density of Ξ has greater or full support in S^{n-1}, the gain in considering directional Lipschitz constants might become negligible.

Lemma 6.4. *Let V be L-smooth. Then, for the three discrete gradient methods and the randomised Itoh–Abe method, the bound*

$$\beta\Big(V(x^k) - \phi_{k+1}\Big) \geq \|\nabla V(x^k)\|^2 \tag{6.5.2}$$

holds for the following values of β.

1. *For the Gonzalez discrete gradient,*

$$\beta = 2\left(\frac{1}{\tau_k} + \frac{1}{2} L^2 \tau_k\right).$$

2. *For the mean value discrete gradient,*

$$\beta = 2\left(\frac{1}{\tau_k} + \frac{1}{4} L^2 \tau_k\right).$$

3. *For the Itoh–Abe discrete gradient,*

$$\beta = 2\left(\frac{1}{\tau_k} + L_{\text{sum}}^2 \tau_k\right).$$

4. *For the randomised Itoh–Abe method,*

$$\beta = \frac{\tau_k}{\zeta}\left(\frac{1}{\tau_k} + \frac{\tilde{L}_{\max}}{2}\right)^2,$$

where ζ is defined in (6.2.7).

Proof. Part 1. By using the characterisation of the Gonzalez discrete gradient in (6.A.1),

$$\overline{\nabla} V(x^k, x^{k+1}) = \left\langle \nabla V\left(\frac{x^k + x^{k+1}}{2}\right), d^\perp \right\rangle d^\perp + \langle \nabla V(z), d\rangle d,$$

where $\langle d, d^\perp\rangle = 0$ and $z \in [x^k, x^{k+1}]$. We calculate, with $\overline{x}^k = (x^k + x^{k+1})/2$,

$$\begin{aligned}
&\|\nabla V(x^k)\|^2 = \langle \nabla V(x^k), d\rangle^2 + \langle \nabla V(x^k), d^\perp\rangle^2 \\
&\leq 2\left(\|\overline{\nabla} V(x^k, x^{k+1})\|^2 + \langle \nabla V(x^k) - \nabla V(z), d\rangle^2 + \left\langle \nabla V(x^k) - \nabla V(\overline{x}^k), d^\perp \right\rangle^2\right) \\
&\leq 2\left(\|\overline{\nabla} V(x^k, x^{k+1})\|^2 + \langle \nabla V(x^k) - \nabla V(z), d\rangle^2 + \frac{1}{4}L^2\|x^k - x^{k+1}\|^2\right).
\end{aligned}$$

Since

$$\langle \nabla V(z), d\rangle = \frac{V(x^{k+1}) - V(x^k)}{\|x^{k+1} - x^k\|}$$

and

$$d = \frac{x^{k+1} - x^k}{\|x^{k+1} - x^k\|},$$

we have

$$\begin{aligned}
\langle \nabla V(x^k) - \nabla V(z), d\rangle^2 &= \frac{\left(\langle \nabla V(x^k), x^{k+1} - x^k\rangle - V(x^{k+1}) + V(x^k)\right)^2}{\|x^k - x^{k+1}\|^2} \\
&\leq \frac{1}{4}L^2\|x^{k+1} - x^k\|^2,
\end{aligned}$$

where the inequality follows from Proposition 6.4 *(i)*. Therefore,

$$\begin{aligned}
\|\nabla V(x^k)\|^2 &\leq 2\left(\|\overline{\nabla} V(x^k, x^{k+1})\|^2 + \langle \nabla V(x^k) - \nabla V(z), d\rangle^2 + \frac{1}{4}L^2\|x^k - x^{k+1}\|^2\right) \\
&\leq 2\left(\frac{1}{\tau_k} + \frac{1}{2}L^2\tau_k\right)\left(V(x^k) - \phi_{k+1}\right),
\end{aligned}$$

where we have used the discrete gradient properties (6.2.5) and (6.2.6).

Part 2. We compute

$$\begin{aligned}\|\nabla V(x^k)\|^2 &\le 2\left(\|\overline{\nabla} V(x^k, x^{k+1})\|^2 + \left\|\int_0^1 \nabla V(sx^k + (1-s)x^{k+1}) - \nabla V(x^k)\,\mathrm{d}s\right\|^2\right)\\ &\le 2\left(\|\overline{\nabla} V(x^k, x^{k+1})\|^2 + \int_0^1 \|\nabla V(sx^k + (1-s)x^{k+1}) - \nabla V(x^k)\|\,\mathrm{d}s\right)^2\\ &\le 2\|\overline{\nabla} V(x^k, x^{k+1})\|^2 + 2L^2\|x^k - x^{k+1}\|^2\left(\int_0^1 s\,\mathrm{d}s\right)^2\\ &= 2\left(\frac{1}{\tau_k} + \frac{1}{4}L^2\tau_k\right)\Big(V(x^k) - \phi_{k+1}\Big).\end{aligned}$$

Part 3. We apply the mean value theorem to

$$\Big(\overline{\nabla} V(x^k, x^{k+1})\Big)_i = \frac{V(x_1^{k+1}, ..., x_i^{k+1}, x_{i+1}^k, ..., x_n^k) - V(x_1^{k+1}, ..., x_{i-1}^{k+1}, x_i^k, ..., x_n^k)}{x_i^{k+1} - x_i^k},$$

to derive that

$$\Big(\overline{\nabla} V(x^k, x^{k+1})\Big)_i = \partial_i V(y^i),$$

where $y^i = [x_1^{k+1}, \ldots, x_{i-1}^{k+1}, c_i, x_{i+1}^k, \ldots, x_n^k]^T$ for some $c_i \in [x_i^k, x_i^{k+1}]$. Therefore, we have

$$\begin{aligned}\|\nabla V(x^k)\|^2 = \sum_{i=1}^n |\partial_i V(x^k)|^2 &\le 2\sum_{i=1}^n |\partial_i V(y^i)|^2 + |\partial_i V(y^i) - \partial_i V(x^k)|^2\\ &\le 2\left(\|\overline{\nabla} V(x^k, x^{k+1})\|^2 + L_{\text{sum}}^2\|x^k - x^{k+1}\|^2\right)\\ &\le 2\left(\frac{1}{\tau_k} + L_{\text{sum}}^2\tau_k\right)\Big(V(x^k) - \phi_{k+1}\Big).\end{aligned}$$

Part 4. By (6.5.1), we have

$$\begin{aligned}\langle \nabla V(x^k), x^k - x^{k+1}\rangle &\le V(x^k) - V(x^{k+1}) + \frac{\overline{L}_{\max}}{2}\|x^k - x^{k+1}\|^2\\ &= \left(\frac{1}{\tau_k} + \frac{\overline{L}_{\max}}{2}\right)\|x^k - x^{k+1}\|^2,\end{aligned}$$

where the second equation follows from (6.2.9).

Furthermore,

$$\langle \nabla V(x^k), x^k - x^{k+1}\rangle = |\langle \nabla V(x^k), d^{k+1}\rangle|\,\|x^k - x^{k+1}\|.$$

From this, we derive

$$\langle \nabla V(x^k), d^{k+1} \rangle^2 \leq \left(\frac{1}{\tau_k} + \frac{\overline{L}_{\max}}{2} \right)^2 \|x^k - x^{k+1}\|^2. \tag{6.5.3}$$

By the definition of ζ, we have

$$\mathbb{E}_{d^{k+1} \sim \Xi} \langle \nabla V(x^k), d^{k+1} \rangle^2 \geq \zeta \|\nabla V(x^k)\|^2. \tag{6.5.4}$$

Combining (6.5.3) and (6.5.4), we derive

$$\|\nabla V(x^k)\|^2 \leq \frac{\tau_k}{\zeta} \left(\frac{1}{\tau_k} + \frac{\overline{L}_{\max}}{2} \right)^2 \Big(V(x^k) - \phi_{k+1} \Big).$$

This concludes the proof. □

6.5.1 Optimal time steps and estimates of β

In Lemma 6.4, lower values for β corresponds to better convergence rates. In what follows, we state the time steps τ_k that yield minimal estimates of β, and how these rates compare to the classical, explicit descent schemes, such as gradient descent and coordinate descent. For the Itoh–Abe discrete gradient method, we are also interested in how β depends on n, i.e. the dimension of the problem.

1. *The Gonzalez discrete gradient method*: The optimal time step and corresponding β are

$$\tau_k = \frac{\sqrt{2}}{L}, \qquad \beta = 2\sqrt{2}L.$$

In comparison, $\beta = 2L$ for explicit gradient descent [30], so this bound is worse by a factor of $\sqrt{2}$.

2. *The mean value discrete gradient method*: The optimal time step and corresponding β are

$$\tau_k = \frac{2}{L}, \qquad \beta = 2L,$$

so in this case, we recover the optimal bound for gradient descent.

3. *The Itoh–Abe discrete gradient method*: The optimal time step and corresponding β are

$$\tau_k = \frac{1}{L_{\text{sum}}}, \qquad \beta = 4L_{\text{sum}} \in [4L, 4\sqrt{n}L].$$

We compare this to the optimal estimates for cyclic coordinate descent schemes in [48, Theorem 3] and [3, Lemma 3.3],

$$\beta = 8\sqrt{n}L,$$

where we have set their parameters $\overline{L}_{\max}$ and $\overline{L}_{\min}$ to $\sqrt{n}L$. The estimate for the Itoh–Abe discrete gradient method is at most half that of the cyclic coordinate descent scheme, even in the worst-case scenario $L_{\text{sum}} = \sqrt{n}L$.

We give one motivating example for considering the parameter L_{sum}. If V is a least squares problem $V(x) = \|Ax - f\|^2/2$, then

$$L_{\text{sum}} \leq \sqrt{\operatorname{rank}(A)}L,$$

so for low-rank system where $\operatorname{rank}(A) \ll n$, the convergence speed of the Itoh–Abe discrete gradient method improves considerably.

To derive that $L_{\text{sum}} \leq \sqrt{\operatorname{rank}(A)}L$, one can show that $L = \|A^*A\|_2$ and $L_{\text{sum}} = \|A^*A\|_F$, where $\|\cdot\|_2$ and $\|\cdot\|_F$ denote the operator norm and the Frobenius norm respectively,

$$\|B\|_2 = \sup_{\|x\|=1} \|Bx\|, \qquad \|B\|_F = \left(\sum_{i=1}^{n} \sum_{j=1}^{n} |b_{ij}|^2 \right)^{1/2},$$

for $B = (b_{ij})_{i,j=1}^n$. The rest follows from the fact that $\|B\|_F \leq \sqrt{\operatorname{rank}(B)}\|B\|_2$ [21, Table 6.2] and that $\operatorname{rank}(A^*A) = \operatorname{rank}(A)$ [26, Statement 4.5.4].

4. *The randomised Itoh–Abe method*: The optimal time step and corresponding β are

$$\tau_k = \frac{2}{\tilde{L}_{\max}}, \qquad \beta = \frac{2}{\zeta}\tilde{L}_{\max}.$$

Recall that when Ξ is the random uniform distribution on the coordinates $(e^i)_{i=1}^n$ or on the unit sphere S^{n-1}, we have $\zeta = 1/n$. This gives us $\beta = 2n\tilde{L}_{\max}$, which, in the former case, is the optimal bound for randomised coordinate descent [48, Equation 30].

6.5.2 Lipschitz continuous gradients

We use the notation $R(x^0) = \operatorname{diam}\left\{x \in \mathbb{R}^n : V(x) \leq V(x^0)\right\}$ throughout the remainder of this section. This is bounded providing V is coercive.

Theorem 6.2. *Let V be an L-smooth, convex, coercive function. Then for all four methods, we have*

$$\phi_k - V^* \leq \frac{\beta R(x^0)^2}{k + 2\frac{\beta}{L}}.$$

where β is given in Lemma 6.4 and V^* is the minimum of V.

Proof. Let x^* be a minimizer of V. By respectively convexity, the Cauchy-Schwarz inequality and Lemma 6.4, we have

$$\begin{aligned}(V(x^k) - V^*)^2 &\leq \left\langle \nabla V(x^k), x^k - x^* \right\rangle^2 \\ &\leq \|\nabla V(x^k)\|^2 \|x^k - x^*\|^2 \\ &\leq \beta R(x^0)^2 (V(x^k) - \phi_{k+1}).\end{aligned}$$

Taking expectation on both sides with respect to ξ_{k-1}, we get

$$(\phi_k - V^*)^2 \leq \beta R(x^0)^2 (\phi_k - \phi_{k+1}).$$

Via the above and by monotonicity of ϕ_k we find that

$$\begin{aligned}\frac{1}{\phi_{k+1} - V^*} - \frac{1}{\phi_k - V^*} &= \frac{\phi_k - \phi_{k+1}}{(\phi_k - V^*)(\phi_{k+1} - V^*)} \\ &\geq \frac{1}{\beta R(x^0)^2} \frac{\phi_k - V^*}{\phi_{k+1} - V^*} \\ &\geq \frac{1}{\beta R(x^0)^2}.\end{aligned}$$

Summing terms from 0 to $k-1$ yields

$$\frac{1}{\phi_k - V^*} - \frac{1}{V(x^0) - V^*} \geq \frac{k}{\beta R(x^0)^2},$$

and, rearranging, we derive

$$\phi_k - V^* \leq \frac{\beta R(x^0)^2}{k + \beta \frac{R(x^0)^2}{V(x^0) - V^*}}.$$

To eliminate dependence on the starting point, we use Proposition 6.4 *(i)*

$$V(x^0) - V^* \leq \frac{L}{2} \|x^0 - x^*\|^2 \leq \frac{L}{2} R(x^0)^2,$$

which gives us

$$\phi_k - V^* \leq \frac{\beta R(x^0)^2}{k + 2\frac{\beta}{L}}.$$

□

6.5.3 The Polyak–Łojasiewicz inequality

The next result shows that for L-smooth functions that satisfy the Polyak–Łojasiewicz (PL) inequality, we achieve a linear convergence rate. The PL inequality is known for extending convergence properties of strongly convex functions to a larger group of functions, including some nonconvex functions. A function is said to satisfy the PL inequality with parameter $\mu > 0$ if, for all $x \in \mathbb{R}^n$,

$$\frac{1}{2}\|\nabla V(x)\|^2 \geq \mu\big(V(x) - V^*\big). \tag{6.5.5}$$

Originally formulated by Polyak in 1963 [36], it was recently shown that this inequality is weaker than other properties commonly used to prove linear convergence [12, 23, 28]. This is useful for extending linear convergence rates to functions that are not strongly convex, including some nonconvex functions.

Proposition 6.5. *Let V be μ-convex. Then V satisfies the PL inequality* (6.5.5) *with parameter μ.*

Proof. See [23]. □

We now proceed to the main result of this subsection.

Theorem 6.3. *Let V be L-smooth and satisfy the PL inequality* (6.5.5) *with parameter μ. Then for the three discrete gradient methods and the randomised Itoh–Abe method, the iterates satisfy*

$$\phi_k - V^* \leq \left(1 - \frac{2\mu}{\beta}\right)^k (V(x^0) - V^*),$$

with β given in Lemma 6.4.

Proof. We combine the PL inequality (6.5.5) with the estimate in Lemma 6.4 to get

$$V(x^k) - \phi_{k+1} \geq \frac{2\mu}{\beta}(V(x^k) - V^*).$$

By taking expectation of both sides with respect to ξ_{k-1}, we obtain

$$\phi_{k+1} - V^* \leq \left(1 - \frac{2\mu}{\beta}\right)(\phi_k - V^*),$$

from which the result follows. □

6.6 Preconditioned discrete gradient method

We briefly discuss the generalisation of the discrete gradient method (6.1.2) to a preconditioned version

$$x^{k+1} = x^k - A_k \overline{\nabla} V(x^k, x^{k+1}), \tag{6.6.1}$$

where $(A_k)_{k\in\mathbb{N}} \subset \mathbb{R}^{n\times n}$ is a sequence of positive-definite matrices. Denoting by $\sigma_{1,k}$ and $\sigma_{n,k}$ the smallest and largest singular values of A_k respectively, we have, for all x,

$$\sigma_{1,k}\|x\| \leq \|A_k x\| \leq \sigma_{n,k}\|x\|.$$

It is straightforward to extend the results in Section 6.3 and Section 6.5 to this setting, under the assumption that there are $\sigma_{\max} \geq \sigma_{\min} > 0$ such that $\sigma_{\min} \leq \sigma_{1,k}, \sigma_{n,k} \leq \sigma_{\max}$ for all $k \in \mathbb{N}$.

We briefly discuss possible motivations for this preconditioning. In the context of geometric integration, it is typical to group the gradient flow system (6.2.1) with the more general dissipative system

$$\dot{x} = -A(x)\nabla V(x),$$

where $A(x) \in \mathbb{R}^{n\times n}$ is positive-definite [38] for all $x \in \mathbb{R}^n$. This yields numerical schemes of the form (6.6.1), where we absorb τ_k into A_k.

There are optimisation problems in which the time step τ_k should vary for each coordinate. This is, for example, the case when one derives the SOR method from the Itoh–Abe discrete gradient method [27]. More generally, if one has coordinate-wise Lipschitz constants for the gradient of the objective function, it may be beneficial to scale the coordinate-wise time steps accordingly.

6.7 Conclusion

In this paper, we studied the discrete gradient method for optimisation, and provided several fundamental results on well-posedness, convergence rates and optimal time steps. We focused on four methods, using the Gonzalez discrete gradient, the mean value discrete gradient, the Itoh–Abe discrete gradient, and a randomised version of the Itoh–Abe method. Several of the proven convergence rates match the optimal rates of classical methods such as gradient descent and stochastic coordinate descent. For the Itoh–Abe discrete gradient method, the proven rates are better than previously established rates for comparable methods, i.e. cyclic coordinate-descent methods [48].

There are open problems to be addressed in future work. First, similar to acceleration for gradient descent and coordinate descent [3, 29, 31, 48], we will

study acceleration of the discrete gradient method to improve the convergence rate from $\mathcal{O}(1/k)$ to $\mathcal{O}(1/k^2)$. Second, we would like to consider generalisations of the discrete gradient method to discretise gradient flow with respect to other measures of distance than the Euclidean inner product, such as Bregman distances [4, 8] and other metrics [5, 9].

Appendix 6.A Bounds on discrete gradients

Lemma 6.5 (Lemma 6.1)**.** *The three discrete gradients satisfy Assumption 6.1 with the following constants.*

1. *For the Gonzalez discrete gradient,*
$$C_n = \sqrt{2}, \qquad \delta \equiv 0.$$
2. *For the mean value discrete gradient,*
$$C_n = 1, \qquad \delta \equiv 0.$$
3. *For the Itoh–Abe discrete gradient,*
$$C_n = \sqrt{n}, \qquad \delta(r) = r.$$

Proof. Case 1. We first consider the Gonzalez discrete gradient. The following characterisation of the Gonzalez discrete gradient will be useful. Denote by d the unit vector
$$d = \frac{y-x}{\|y-x\|}.$$
Then there is a vector $d^\perp$ such that $\langle d, d^\perp \rangle = 0$, $\|d^\perp\| = 1$, and
$$\nabla V\left(\frac{x+y}{2}\right) = \left\langle \nabla V\left(\frac{x+y}{2}\right), d \right\rangle d + \left\langle \nabla V\left(\frac{x+y}{2}\right), d^\perp \right\rangle d^\perp.$$
We rewrite the Gonzalez discrete gradient as
$$\overline{\nabla} V(x,y) = \left\langle \nabla V\left(\frac{x+y}{2}\right), d^\perp \right\rangle d^\perp + \frac{V(y)-V(x)}{\|y-x\|} d.$$
By the mean value theorem, there is $z \in [x, y]$ such that
$$V(y) - V(x) = \langle \nabla V(z), y - x \rangle.$$
Therefore, we obtain
$$\overline{\nabla} V(x,y) = \left\langle \nabla V\left(\frac{x+y}{2}\right), d^\perp \right\rangle d^\perp + \langle \nabla V(z), d \rangle d. \tag{6.A.1}$$

From this, we derive

$$\|\overline{\nabla} V(x,y)\|^2 \leq \left\| \nabla V\left(\frac{x+y}{2}\right)\right\|^2 + \left\|\nabla V(z)\right\|^2.$$

This implies that properties *(i)* and *(ii)* hold with $C_n = \sqrt{2}$ and $\delta \equiv 0$. To show property *(iii)*, it is sufficient to note that since K is convex and has nonempty interior, $\nabla W\left((x+y)/2\right) = \nabla V\left((x+y)/2\right)$.

Case 2. Next we consider the mean value discrete gradient. It is clear that properties *(i)* and *(ii)* hold with $C_n = 1$ and $\delta \equiv 0$. To show property *(iii)*, it is sufficient to note that since K is convex and has nonempty interior, $\nabla W(z) = \nabla V(z)$ for all $z \in [x,y]$.

Case 3. For the Itoh–Abe discrete gradient, we set $\delta(r) = r$. By applying the mean value theorem to

$$\left(\overline{\nabla} V(x,y)\right)_i = \frac{V(y_1,\dots,y_i,x_{i+1},\dots,x_n) - V(y_1,\dots,y_{i-1},x_i,\dots,x_n)}{y_i - x_i},$$

we derive that

$$\left(\overline{\nabla} V(x,y)\right)_i = \partial_i V(z^i),$$

where $z^i = [x_1^{k+1},\dots,x_{i-1}^{k+1},c_i,x_{i+1}^k,\dots,x_n^k]^T$ for some $c_i \in [x_i^k, x_i^{k+1}]$. Furthermore, we have

$$\|z^i - x\| \leq \|y - x\|,$$

so $z \in K_{\mathrm{diam}(K)}$. This implies that properties *(i)* and *(ii)* hold with $C_n = \sqrt{n}$. Property *(iii)* is immediate. □

Appendix 6.B Cutoff function

We provide proof of existence of an appropriate cutoff function in Theorem 6.1 part *(iii)*. While this is based on standard arguments using mollifiers, the authors could not find a result in the literature specifically for cutoff functions with noncompact support and controlled derivatives. We therefore include one for completeness.

Lemma 6.6. *Let $V \subset U \subset \mathbb{R}^n$ be sets such that for some $\varepsilon > 0$,*

$$\mathrm{dist}(V, \mathbb{R}^n \setminus U) := \inf_{x \in V, y \notin U} \|x - y\| \geq \varepsilon.$$

Then there is a cutoff function $\varphi \in C^\infty(\mathbb{R}^n; [0,1])$ such that

$$\varphi(x) = \begin{cases} 1 & \text{if } x \in V, \\ 0 & \text{if } x \notin U, \end{cases}$$

and such that $\nabla\varphi$ is uniformly bounded on $\mathbb{R}^n$.

Proof. We will construct a cutoff function with a uniformly bounded gradient. Denote by W the set $\mathbb{R}^n \setminus U$ and consider the distance functions

$$d_V(x) := \inf_{z\in V} \|x - z\|, \qquad d_W(x) := \inf_{z\in W} \|x - z\|.$$

For any $x \in V$, $y \in W$ and $z \in \mathbb{R}^n$,

$$\varepsilon \le \|x - y\| \le \|x - z\| + \|y - z\|.$$

Taking the infimum over all $z \in V$ and $w \in W$, we deduce that

$$d_V(x) + d_W(x) \ge \varepsilon. \tag{6.B.1}$$

Let $\psi : \mathbb{R}^n \to [0, 1]$ be defined by

$$\psi(x) := \frac{d_W(x)}{d_V(x) + d_W(x)}.$$

This function satisfies $\psi(X) = 1$ for $x \in V$, $\phi(x) = 0$ for $x \in W$ and $\psi(x) \in [0, 1]$ otherwise. We show that it is Lipschitz continuous with Lipschitz constant $1/\varepsilon$.

$$\begin{aligned}
|\psi(x) - \psi(y)| &= \left| \frac{d_W(x)}{d_V(x) + d_W(x)} - \frac{d_W(y)}{d_V(y) + d_W(y)} \right| \\
&= \left| \frac{\big(d_V(y) - d_V(x)\big)\, d_W(x) + \big(d_W(x) - d_W(y)\big)\, d_V(x)}{\big(d_W(x) + d_V(x)\big)\big(d_W(y) + d_V(y)\big)} \right| \\
&\le \frac{\Big|\big(d_V(y) - d_V(x)\big)\, d_W(x)\Big| + \Big|\big(d_W(x) - d_W(y)\big)\, d_V(x)\Big|}{\Big|\big(d_W(x) + d_V(x)\big)\big(d_W(y) + d_V(y)\big)\Big|} \\
&\le \frac{1}{\varepsilon}\|x - y\| \left(\left| \frac{d_W(x)}{d_W(x) + d_V(x)} \right| + \left| \frac{d_V(x)}{d_W(x) + d_V(x)} \right| \right) \\
&= \frac{1}{\varepsilon}\|x - y\|.
\end{aligned}$$

The second inequality above follows from (6.B.1) and the Lipschitz continuity of d_V, d_W,

$$|d_V(x) - d_V(y)| \le \|x - y\|, \qquad |d_W(x) - d_W(y)| \le \|x - y\|.$$

So ψ is a Lipschitz continuous cutoff function for V and W with Lipschitz constant $1/\varepsilon$.

We choose an appopriate mollifier $J \in \mathcal{C}_c^\infty(\mathbb{R}^n; [0, \infty))$ such that $\int_{\mathbb{R}^n} J(x)\,\mathrm{d}x = 1$ and $J(x) = 0$ whenever $\|x\| \ge \varepsilon/2$, and convolve it with ψ. It is easy to check that the resultant function,

$$\varphi(x) = \int_{\mathbb{R}^n} J(z)\psi(x - z)\,\mathrm{d}z,$$

is in $\mathcal{C}^\infty(\mathbb{R}^n; [0,1])$ and satisfies

$$\varphi(x) = \begin{cases} 1 & \text{if } x \in V, \\ 0 & \text{if } x \notin U. \end{cases}$$

This is a standard result, see for example [1, Theorem 2.29]. To conclude, we show that $\|\nabla\varphi(x)\| \leq 1/\varepsilon$. We do so by showing that φ inherits the Lipschitz continuity of ψ with the same Lipschitz constant, being $1/\varepsilon$. We have

$$\begin{aligned} |\varphi(x) - \varphi(y)| &\leq \int_{\mathbb{R}^n} \big|\psi(x-z) - \psi(y-z)\big| \, |J(z)| \, \mathrm{d}z \\ &\leq \frac{1}{\varepsilon} \|x-y\| \int_{\mathbb{R}^n} |J(z)| \, \mathrm{d}z \\ &= \frac{1}{\varepsilon} \|x-y\|. \end{aligned}$$

This concludes the proof. □

Bibliography

[1] R. A. Adams and J. J. F. Fournier, *Sobolev spaces*, vol. 140 of Pure and Applied Mathematics (Amsterdam), Elsevier/Academic Press, Amsterdam, second ed., 2003.

[2] L. Ambrosio, N. Gigli, and G. Savaré, *Gradient flows in metric spaces and in the space of probability measures*, Lectures in Mathematics ETH Zürich, Birkhäuser Verlag, Basel, second ed., 2008.

[3] A. Beck and L. Tetruashvili, *On the convergence of block coordinate descent type methods*, SIAM J. Optimiz., 23 (2013), pp. 2037–2060.

[4] M. Benning, M. M. Betcke, M. J. Ehrhardt, and C.-B. Schönlieb, *Choose your path wisely: gradient descent in a Bregman distance framework*, arXiv preprint arXiv:1712.04045, (2017).

[5] M. Benning, L. Calatroni, B. Düring, and C.-B. Schönlieb, *A primal-dual approach for a total variation Wasserstein flow*, in Geometric science of information, vol. 8085 of Lecture Notes in Comput. Sci., Springer, Heidelberg, 2013, pp. 413–421.

[6] D. P. Bertsekas, *Nonlinear programming*, Athena Scientific Optimization and Computation Series, Athena Scientific, Belmont, MA, second ed., 1999.

[7] L. E. J. BROUWER, *über Abbildung von Mannigfaltigkeiten*, Math. Ann., 71 (1911), pp. 97–115.

[8] M. BURGER, G. GILBOA, S. OSHER, AND J. XU, *Nonlinear inverse scale space methods*, Commun. Math. Sci., 4 (2006), pp. 179–212.

[9] M. BURGER, L. HE, AND C.-B. SCHÖNLIEB, *Cahn-Hilliard inpainting and a generalization for grayvalue images*, SIAM J. Imaging Sci., 2 (2009), pp. 1129–1167.

[10] E. CELLEDONI, S. EIDNES, B. OWREN, AND T. RINGHOLM, *Dissipative schemes on Riemannian manifolds*, arXiv preprint arXiv:1804.08104, (2018).

[11] E. CELLEDONI, V. GRIMM, R. I. MCLACHLAN, D. I. MCLAREN, D. O'NEALE, B. OWREN, AND G. R. W. QUISPEL, *Preserving energy resp. dissipation in numerical PDEs using the "average vector field" method*, J. Comput. Phys., 231 (2012), pp. 6770–6789.

[12] D. CSIBA AND P. RICHTÁRIK, *Global convergence of arbitrary-block gradient methods for generalized Polyak-Łojasiewicz functions*, arXiv preprint arXiv:1709.03014, (2017).

[13] M. EHRHARDT, G. QUISPEL, E. RIIS, AND C.-B. SCHÖNLIEB, *A geometric integration approach to nonsmooth, nonconvex optimization*, In preparation, (2018).

[14] O. FERCOQ AND P. RICHTÁRIK, *Accelerated, parallel, and proximal coordinate descent*, SIAM J. Optim., 25 (2015), pp. 1997–2023.

[15] O. GONZALEZ, *Time integration and discrete Hamiltonian systems*, J. Nonlinear Sci., 6 (1996), pp. 449–467.

[16] V. GRIMM, R. I. MCLACHLAN, D. I. MCLAREN, G. QUISPEL, AND C. SCHÖNLIEB, *Discrete gradient methods for solving variational image regularisation models*, J. Phys. A: Math. Theor., 50 (2017).

[17] E. HAIRER AND C. LUBICH, *Energy-diminishing integration of gradient systems*, IMA J. Numer. Anal., 34 (2013), pp. 452–461.

[18] E. HAIRER, C. LUBICH, AND G. WANNER, *Geometric numerical integration*, vol. 31 of Springer Series in Computational Mathematics, Springer, Heidelberg, 2010.

[19] A. HARTEN, P. D. LAX, AND B. VAN LEER, *On upstream differencing and Godunov-type schemes for hyperbolic conservation laws*, SIAM Rev., 25 (1983), pp. 35–61.

[20] Y. HERNÁNDEZ-SOLANO, M. ATENCIA, G. JOYA, AND F. SANDOVAL, *A discrete gradient method to enhance the numerical behaviour of Hopfield networks*, Neurocomputing, 164 (2015), pp. 45–55.

[21] N. J. HIGHAM, *Accuracy and stability of numerical algorithms*, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, second ed., 2002.

[22] T. ITOH AND K. ABE, *Hamiltonian-conserving discrete canonical equations based on variational difference quotients*, J. Comput. Phys., 76 (1988), pp. 85–102.

[23] H. KARIMI, J. NUTINI, AND M. SCHMIDT, *Linear convergence of gradient and proximal-gradient methods under the Polyak–Łojasiewicz condition*, in ECML PKDD, Springer, 2016, pp. 795–811.

[24] R. MCLACHLAN AND R. QUISPEL, *Six lectures on the geometric integration of ODEs*, in Foundations of computational mathematics (Oxford, 1999), vol. 284 of London Math. Soc. Lecture Note Ser., Cambridge Univ. Press, Cambridge, 2001, pp. 155–210.

[25] R. I. MCLACHLAN, G. R. W. QUISPEL, AND N. ROBIDOUX, *Geometric integration using discrete gradients*, Philos. T. R. Soc. A, 357 (1999), pp. 1021–1045.

[26] C. MEYER, *Matrix analysis and applied linear algebra*, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2000.

[27] Y. MIYATAKE, T. SOGABE, AND S.-L. ZHANG, *On the equivalence between SOR-type methods for linear systems and discrete gradient methods for gradient systems*, arXiv preprint arXiv:1711.02277, (2017).

[28] I. NECOARA, Y. NESTEROV, AND F. GLINEUR, *Linear convergence of first order methods for non-strongly convex optimization*, Math. Program., (2018), pp. 1–39.

[29] Y. NESTEROV, *A method of solving a convex programming problem with convergence rate* $\mathcal{O}(1/k^2)$, Sov. Math. Dokl., 27 (1983), pp. 372–376.

[30] Y. NESTEROV, *Introductory lectures on convex optimization*, vol. 87 of Applied Optimization, Kluwer Academic Publishers, Boston, MA, 2004.

[31] Y. NESTEROV, *Efficiency of coordinate descent methods on huge-scale optimization problems*, SIAM J. Optim, 22 (2012), pp. 341–362.

[32] Y. NESTEROV AND V. SPOKOINY, *Random gradient-free minimization of convex functions*, Found. Comput. Math., 17 (2017), pp. 527–566.

[33] J. NESTRUEV, *Smooth manifolds and observables*, vol. 220 of Graduate Texts in Mathematics, Springer-Verlag, New York, 2003.

[34] J. NOCEDAL AND S. WRIGHT, *Numerical Optimization*, Springer Series in Operations Research and Financial Engineering, Springer-Verlag, New York, 2 ed., 2006.

[35] R. A. NORTON AND G. R. W. QUISPEL, *Discrete gradient methods for preserving a first integral of an ordinary differential equation*, Discrete Contin. Dyn. Syst., 34 (2014), pp. 1147–1170.

[36] B. T. POLJAK, *Gradient methods for minimizing functionals*, Ž. Vyčisl. Mat. i Mat. Fiz., 3 (1963), pp. 643–653.

[37] Z. QU, P. RICHTÁRIK, AND T. ZHANG, *Quartz: Randomized dual coordinate ascent with arbitrary sampling*, in Adv. Neur. In. 28, 2015, pp. 865–873.

[38] G. R. W. QUISPEL AND G. S. TURNER, *Discrete gradient methods for solving ODEs numerically while preserving a first integral*, J. Phys. A, 29 (1996), pp. L341–L349.

[39] T. RINGHOLM, J. LAZIĆ, AND C.-B. SCHÖNLIEB, *Variational image regularization with Euler's elastica using a discrete gradient scheme*, arXiv preprint arXiv:1712.07386, (2017).

[40] W. RUDIN, *Principles of mathematical analysis*, McGraw-Hill Book Co., New York-Auckland-Düsseldorf, third ed., 1976.

[41] F. SANTAMBROGIO, *{Euclidean, metric, and Wasserstein} gradient flows: an overview*, Bull. Math. Sci., 7 (2017), pp. 87–154.

[42] J. SCHAUDER, *Der Fixpunktsatz in Funktionalraümen*, Studia Mathematica, 2 (1930), pp. 171–180.

[43] D. SCIEUR, V. ROULET, F. BACH, AND A. D'ASPREMONT, *Integration methods and optimization algorithms*, in Adv. Neur. In. 30, 2017, pp. 1109–1118.

[44] S. STICH, *Convex optimizaiton with randum pursuit*, PhD thesis, ETH Zurich, 2014.

[45] A. M. STUART AND A. R. HUMPHRIES, *Dynamical systems and numerical analysis*, vol. 2 of Cambridge Monographs on Applied and Computational Mathematics, Cambridge University Press, Cambridge, 1996.

[46] W. SU, S. BOYD, AND E. J. CANDÈS, *A differential equation for modeling Nesterov's accelerated gradient method: theory and insights*, J. Mach. Learn. Res., 17 (2016), pp. Paper No. 153, 43.

[47] A. WIBISONO, A. C. WILSON, AND M. I. JORDAN, *A variational perspective on accelerated methods in optimization*, Proc. Natl. Acad. Sci. USA, 113 (2016), pp. E7351–E7358.

[48] S. J. WRIGHT, *Coordinate descent algorithms*, Math. Program., 151 (2015), pp. 3–34.

Dissipative numerical schemes on Riemannian manifolds with applications to gradient flows

Elena Celledoni, Sølve Eidnes, Brynjulf Owren and Torbjørn Ringholm

Submitted

Dissipative numerical schemes on Riemannian manifolds with applications to gradient flows

Abstract. This paper concerns an extension of discrete gradient methods to finite-dimensional Riemannian manifolds termed discrete Riemannian gradients, and their application to dissipative ordinary differential equations. This includes Riemannian gradient flow systems which occur naturally in optimization problems. The Itoh–Abe discrete gradient is formulated and applied to gradient systems, yielding a derivative-free optimization algorithm. The algorithm is tested on two eigenvalue problems and two problems from manifold valued imaging: InSAR denoising and DTI denoising.

7.1 Introduction

When designing and applying numerical schemes for solving systems of ODEs and PDEs there are several important properties which serve to distinguish schemes, one of which is the preservation of geometric features of the original system. The field of geometric integration encompasses many types of numerical schemes for ODEs and PDEs specifically designed to preserve one or more such geometric features; a non-exhaustive list of features includes symmetry, symplecticity, first integrals (or energy), orthogonality, and manifold structures such as Lie group structure [13]. Energy conserving methods have a successful history in the field of numerical integration of ODEs and PDEs. In a similar vein, numerical schemes with guaranteed dissipation are useful for solving dissipative equations such as gradient systems.

As seen in [15], any Runge–Kutta method can be dissipative when applied to gradient systems as long as step sizes are chosen small enough; less severe but still restrictive conditions for dissipation in Runge–Kutta methods are presented in [12]. In [9], Gonzalez introduces the notion of discrete gradient schemes with energy preserving properties, later expanded upon to include dissipative systems in [19]. These articles consider ODEs in Euclidian spaces only. Unlike the Runge–Kutta methods, discrete gradient methods are dissipative for all step sizes, meaning one can employ adaptive time steps while retaining convergence toward fixed points [23]. Motivated by their work on Lie group methods, the energy conserving discrete gradient method was generalized to ODEs on manifolds, and Lie groups particularly, in [6] where the authors introduce the concept of discrete differentials. In [5], this concept is specialized in the setting of Riemannian manifolds. To the best of our knowledge, the

This work was supported by the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 691070.

discrete gradient methods have not yet been formulated for dissipative ODEs on manifolds. Doing so is the central purpose of this article.

One of the main reasons for generalizing discrete gradient methods to dissipative systems on manifolds is that gradient systems are dissipative, and gradient flows are natural tools for optimization problems which arise in e.g. manifold-valued image processing and eigenvalue problems. The goal is then to find one or more stationary points of the gradient flow of a functional $V : M \to \mathbb{R}$, which correspond to critical points of V. This approach is, among other optimization methods, presented in [1]. Since gradient systems occur naturally on Riemannian manifolds, it is natural to develop our schemes in a Riemannian manifold setting.

A similarity between the optimization algorithms in [1] and the manifold valued discrete gradient methods in [6] is their use of retraction mappings. Retraction mappings were introduced for numerical methods in [24], see also [2]; they are intended as computationally efficient alternatives to parallel transport on manifolds. Our methods will be formulated as a framework using general discrete gradients on general Riemannian manifolds with general retractions. We will consider a number of specific examples that illustrate how to apply the procedure in practical problems.

As detailed in [10] and [20], using the Itoh–Abe discrete gradient [16], one can obtain an optimization scheme for n-dimensional problems with a limited degree of implicitness. At every iteration, one needs to solve n decoupled scalar nonlinear subequations, amounting to $\mathcal{O}(n)$ operations per step. In other discrete gradient schemes a system of n coupled nonlinear equations must be solved per iteration, amounting to $\mathcal{O}(n^2)$ operations per step. The Itoh–Abe discrete gradient method therefore appears to be well suited to large-scale problems such as image analysis problems, and so it seems natural to apply our new methods to image analysis problems on manifolds, see section 7.4.2. In [6], the authors generalize the average vector field [14] and midpoint [9] discrete gradients, but not the Itoh–Abe discrete gradient, to Lie groups and homogeneous manifolds. A novelty of this article is the formulation of the Itoh–Abe discrete gradient for problems on manifolds.

As examples we will consider two eigenvalue finding problems, in addition to the more involved problems of denoising InSAR and DTI images using total variation (TV) regularization [28]. The latter two problems we consider as real applications of the algorithm. The two eigenvalue problems are included mostly for the exposition and illustration of our methods, as well as for testing convergence properties.

The paper is organized as follows: Below, we introduce notation and fix some fundamental definitions used later on. In the next section, we formulate the dissipative problems we wish to solve. In section 3, we present the discrete

Riemannian gradient (DRG) methods, a convergence proof for the family of optimization methods obtained by applying DRG methods to Riemannian gradient flow problems, the Itoh–Abe discrete gradient generalized to manifolds, and the optimization algorithm obtained by applying the Itoh–Abe DRG to the gradient flow problem. In section 4, we provide numerical experiments to illustrate the use of DRGs in optimization, and in the final section we present conclusions and avenues for future work.

Notation and preliminaries

Some notation and definitions used in the following are summarized below. For a more thorough introduction to the concepts, see e.g. [17] or [18].

Notation	Description
M	n-dimensional Riemannian manifold
T_pM	tangent space at $p \in M$ with zero vector 0_p
T_p^*M	cotangent space at $p \in M$
TM	tangent bundle of M
T^*M	cotangent bundle of M
$\mathfrak{X}(M)$	space of vector fields on M
$g(\cdot,\cdot)$	Riemannian metric on M
$\|\cdot\|_p$	Norm induced on T_pM by g
$\{E_l\}_{l=1}^n$	g-orthogonal basis of T_pM

On any differentiable manifold there is a duality pairing $\langle\cdot,\cdot\rangle : T^*M \times TM \to \mathbb{R}$ which we will denote as $\langle\omega, v\rangle = \omega(v)$. Furthermore, the Riemannian metric sets up an isomorphism between TM and T^*M via the linear map $v \mapsto g(v,\cdot)$. This map and its inverse, termed the musical isomorphisms, are known as the flat map $^\flat : TM \to T^*M$ and sharp map $^\sharp : T^*M \to TM$, respectively. The applications of these maps are also termed index raising and lowering when considering the tensorial representation of the Riemannian metric. Note that with the above notation we have the idiom $x^\flat(y) = \left\langle x^\flat, y \right\rangle = g(x,y)$.

On a Riemannian manifold, one can define gradients: For $V \in C^\infty(M)$, the (Riemannian) gradient with respect to g, $\operatorname{grad}_g V \in \mathfrak{X}(M)$, is the unique vector field such that $g(\operatorname{grad}_g V, X) = \langle \mathrm{d}V, X \rangle$ for all $X \in \mathfrak{X}(M)$. In the language of musical isomorphisms, $\operatorname{grad}_g V = (\mathrm{d}V)^\sharp$. For the remainder of this article, we will write $\operatorname{grad} V$ for the gradient and assume that it is clear from the context which g is to be used.

Furthermore, the *geodesic* between p and q is the unique curve of minimal length between p and q, providing a distance function $d_M : M \times M \to \mathbb{R}$. The geodesic γ passing through p with tangent v is given by the Riemannian exponential at p, $\gamma(t) = \exp_p(tv)$. For any p, $\exp_p$ is a diffeomorphism on

a neighbourhood N_p of 0_p, The image $\exp_p(S_p)$ of any star-shaped subset $S_p \subset N_p$ is called a normal neighbourhood of p, and on this, $\exp_p$ is a radial isometry, i.e. $d_M(\exp_p(u), \exp_p(v)) = g(u, v)$ for all $u, v \in S_p$.

7.2 The problem

We will consider ordinary differential equations (ODEs) of the form

$$\dot{u} = F(u), \quad u(0) = u^0 \in M, \tag{7.2.1}$$

where $F \in \mathfrak{X}(M)$ has an associated energy $V : M \to \mathbb{R}$ dissipating along solutions of (7.2.1). That is, with $u(t)$ a solution of (7.2.1):

$$\frac{\mathrm{d}}{\mathrm{d}t} V(u) = \left\langle \mathrm{d}V(u), \dot{u} \right\rangle = \left\langle \mathrm{d}V(u), F(u) \right\rangle = g(\mathrm{grad}V(u), F(u) \leq 0.$$

An example of such an ODE is the gradient flow. Given an energy V, the gradient flow of V with respect to a Riemannian metric g is

$$\dot{u} = -\mathrm{grad}V(u), \tag{7.2.2}$$

which is dissipative since if $u(t)$ solves (7.2.2), we have

$$\frac{\mathrm{d}}{\mathrm{d}t} V(u) = -g\left(\mathrm{grad}V(u), \mathrm{grad}V(u)\right) \leq 0.$$

This can be generalized slightly by an approach similar to that in [19]. Suppose there exists a (0,2) tensor field h on M such that $h(x, x) \leq 0$. We can associate to h the (1,1) tensor field $H : TM \to TM$ given by $Hx = h(x, \cdot)^\sharp$. Now, consider the system

$$\dot{u} = H\mathrm{grad}V(u). \tag{7.2.3}$$

This system dissipates V, since

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t} V(u) &= \left\langle \mathrm{d}V(u), \dot{u} \right\rangle \\ &= \left\langle \mathrm{d}V(u), H\mathrm{grad}V(u) \right\rangle \\ &= g\left(\mathrm{grad}V(u), H\mathrm{grad}V(u)\right) \\ &= h\left(\mathrm{grad}V(u), \mathrm{grad}V(u)\right) \leq 0. \end{aligned}$$

Any dissipative system of the form (7.2.1) can be written in this form on the set $M \backslash \{p \in M : g(F(p), \mathrm{grad}V(p)) = 0\}$ since, given F and V, we can construct h as follows:

$$h = \frac{1}{g(F, \mathrm{grad}V)} F^\flat \otimes F^\flat.$$

If $F = -\mathrm{grad}V$, we take $h = -g$ such that H becomes the negative identity, and recover (7.2.2).

7.3 Numerical scheme

The discrete differentials in [6] are formulated such that they may be used on non-Riemannian manifolds. Since we restrict ourselves to Riemannian manifolds, we may define their Riemannian analogues: discrete Riemannian gradients. As with the discrete differentials, we shall make use of retractions as defined in [24].

Definition 7.1. Let $\phi : TM \to M$ and denote by ϕ_p the restriction of ϕ to T_pM. Then, ϕ is a *retraction* if the following conditions are satisfied:

- ϕ_p is smooth and defined in an open ball $B_{r_p(0_p)}$ of radius r_p around 0_p, the zero vector in T_pM.
- $\phi_p(v) = p$ if and only if $v = 0_p$.
- Identifying $T_{0_p}T_pM \simeq T_pM$, ϕ_p satisfies

$$d\phi_p\big|_{0_p} = \mathrm{id}_{T_pM},$$

where id_{T_pM} denotes the identity mapping on T_pM.

From the inverse function theorem it follows that for any p, there exists a neighbourhood $U_{p,\phi} \in T_pM$ of 0_p, such that $\phi_p : U_{p,\phi} \to \phi_p(U_{p,\phi})$ is a diffeomorphism. In general, ϕ_p is not a diffeomorphism on the entirety of T_pM and so all the following schemes must be considered local in nature. The canonical retraction on a Riemannian manifold is the Riemannian exponential. This may be computationally expensive to evaluate even if closed expressions for geodesics are known, and so one often wishes to come up with less costly retractions if possible. We are now ready to introduce the notion of discrete Riemannian gradients.

Definition 7.2. Given a retraction ϕ, a function $c : M \times M \to M$ where $c(p, p) = p$ for all $p \in M$ and a continuous $V : M \to \mathbb{R}$, then $\overline{\mathrm{grad}}V : M \times M \to TM$ is a discrete Riemannian gradient of V if it is continuous and, for all $p, q \in U_{c(p,q),\phi}$,

$$V(q) - V(p) = g\left(\overline{\mathrm{grad}}V(p, q), \phi^{-1}_{c(p,q)}(q) - \phi^{-1}_{c(p,q)}(p)\right) \tag{7.3.1}$$

$$\overline{\mathrm{grad}}V(p, p) = \mathrm{grad}V|_p. \tag{7.3.2}$$

We formulate a numerical scheme for equation (7.2.3) based on this definition. Given times $0 = t_0 < t_1 < ...$, let u^k denote the approximation to $u(t_k)$ and let $\tau_k = t_{k+1} - t_k$. Then, we take

$$u^{k+1} = \phi_{c^k}\left(W(u^k, u^{k+1})\right) \tag{7.3.3}$$

$$W(u^k, u^{k+1}) = \phi^{-1}_{c^k}(u^k) - \tau_k \overline{H}_{(u^k,u^{k+1})}\overline{\mathrm{grad}}V(u^k, u^{k+1}) \tag{7.3.4}$$

where $c^k = c(u^k, u^{k+1})$ and $\overline{H}_{(p,q)}$ is the (1,1) tensor associated with a negative semi-definite (0,2) tensor field $\overline{h}_{(p,q)} : T_{c(p,q)}M \times T_{c(p,q)}M \to \mathbb{R}$ approximating $h|_p$ consistently, that is:

$$\begin{aligned}\overline{h}_{(p,p)}(v, w) &= h|_p(v, w)\\ \overline{h}_{(p,q)}(u, u) &\leq 0.\end{aligned}$$

In the above and all of the following, we assume that u^k and u^{k+1} lie in $U_{c^k,\phi} \cap S_{c^k}$. The following proposition verifies that the scheme is dissipative.

Proposition 7.1. *: The sequence $\{u^k\}_{k\in\mathbb{N}}$ generated by the DRG scheme (7.3.3)-(7.3.4) satisfies $V(u^{k+1}) - V(u^k) \leq 0$ for all $k \in \mathbb{N}$.*

Proof. Using property (7.3.1) and equations (7.3.3) and (7.3.4), we get

$$\begin{aligned}V(u^{k+1}) - V(u^k) &= g\left(\overline{\operatorname{grad}}V(u^k, u^{k+1}), \phi^{-1}_{c^k}(u^{k+1}) - \phi^{-1}_{c^k}(u^k)\right)\\ &= g\left(\overline{\operatorname{grad}}V(u^k, u^{k+1}), W(u^k, u^{k+1}) - \phi^{-1}_{c^k}(u^k)\right)\\ &= -\tau_k g\left(\overline{\operatorname{grad}}V(u^k, u^{k+1}), \overline{H}_{(u^k,u^{k+1})}\overline{\operatorname{grad}}V(u^k, u^{k+1}))\right)\\ &= -\tau_k \bar{h}_{(u^k,u^{k+1})}\left(\overline{\operatorname{grad}}V(u^k, u^{k+1}), \overline{\operatorname{grad}}V(u^k, u^{k+1})\right) \leq 0. \quad \square\end{aligned}$$

Two DRGs, the AVF DRG and the Gonzalez DRG, can be easily found by index raising the discrete differentials defined in [6]. We will later generalize the Itoh–Abe discrete gradient, but first we present a proof that the DRG scheme converges to a stationary point when used as an optimization method. We will need the following definition of coercivity:

Definition 7.3. A function $V : M \to \mathbb{R}$ is *coercive* if, for all $v \in M$, every sequence $\{u^k\}_{k\in\mathbb{N}} \subset M$ such that $\lim_{k\to\infty} d_M(u^k, v) = \infty$, satisfies $\lim_{k\to\infty} V(u^k) = \infty$.

We will also need the following theorem from [26], concerning the boundedness of the sublevel sets $M_\mu = \{u \in M : V(u) \leq \mu\}$ of V:

Theorem 7.1. *Assume M is unbounded. Then the sublevel sets of $V : M \to \mathbb{R}$ are bounded if and only if V is coercive.*

Proof. See [26], Theorem 8.6, Chapter 1 and the remarks below it.

Equipped with this, we present the following theorem, the proof of which is inspired by that of the convergence theorem in [10].

Theorem 7.2. *Assume that M is geodesically complete, that $V : M \to R$ is coercive, bounded from below and continuously differentiable, and that $\overline{\text{grad}}V$ is continuous. Then, the iterates $\{u^k\}_{k\in\mathbb{N}}$ produced by applying the discrete Riemannian gradient scheme (7.3.3)-(7.3.4) with time steps $0 < \tau_{min} \leq \tau_k \leq \tau_{max}$ and $c^k = u^k$ or $c^k = u^{k+1}$, to the gradient flow of V satisfy*

$$\lim_{k\to\infty} \overline{\text{grad}}V(u^k, u^{k+1}) = \lim_{k\to\infty} \text{grad}V(u^k) = 0.$$

Additionally, there exists at least one accumulation point u^* of $\{u^k\}_{k\in\mathbb{N}}$, and any such accumulation point satisfies $\text{grad}V(u^*) = 0$.

Proof. Since V is bounded from below and by Proposition 7.1, we have

$$C \leq V(u^{k+1}) \leq V(u^k) \leq ... \leq V(u^0)$$

such that, by the monotone convergence theorem, $V^* := \lim_{k\to\infty} V(u^k)$ exists. Furthermore, by property (7.3.1) and using the scheme (7.3.3)-(7.3.4):

$$\begin{aligned}\frac{1}{\tau_k}\left\|\phi_{c^k}^{-1}(u^k) - \phi_{c^k}^{-1}(u^{k+1})\right\|_{c^k}^2 &= \tau_k \left\|\overline{\text{grad}}V(u^k, u^{k+1})\right\|_{c^k}^2 \\ &= g\left(\overline{\text{grad}}V(u^k, u^{k+1}), \phi_{c^k}^{-1}(u^k) - \phi_{c^k}^{-1}(u^{k+1})\right) \\ &= V(u^k) - V(u^{k+1}).\end{aligned}$$

From this, it is clear that for any $i, j \in \mathbb{N}$,

$$\sum_{k=i}^{j-1} \tau_k \left\|\overline{\text{grad}}V(u^k, u^{k+1})\right\|_{c^k}^2 = V(u^i) - V(u^j) \leq V(u^0) - V^*$$

and

$$\sum_{k=i}^{j-1} \frac{1}{\tau_k}\left\|\phi_{c^k}^{-1}(u^k) - \phi_{c^k}^{-1}(u^{k+1})\right\|_{c^k}^2 = V(u^i) - V(u^j) \leq V(u^0) - V^*.$$

In particular,

$$\sum_{k=0}^{\infty} \left\|\overline{\text{grad}}V(u^k, u^{k+1})\right\|_{c^k}^2 \leq \frac{V(u^0) - V^*}{\tau_{min}},$$

and

$$\sum_{k=0}^{\infty} \left\|\phi_{c^k}^{-1}(u^k) - \phi_{c^k}^{-1}(u^{k+1})\right\|_{c^k}^2 \leq \tau_{max}\left(V(u^0) - V^*\right),$$

meaning

$$\lim_{k\to\infty} \left\| \overline{\operatorname{grad}} V(u^k, u^{k+1}) \right\|_{c^k} = 0,$$
$$\lim_{k\to\infty} \left\| \phi_{c^k}^{-1}(u^k) - \phi_{c^k}^{-1}(u^{k+1}) \right\|_{c^k} = 0.$$

Since u^{k+1} is in a normal neighbourhood of c^k,

$$d_M(c^k, u^{k+1}) = d_M(c^k, \exp_{c^k}(\exp_{c^k}^{-1}(u^{k+1}))) = \| \exp_{c^k}^{-1}(u^{k+1}) \|_{c^k}. \tag{7.3.5}$$

Introduce $\psi_{c^k} : T_{c^k}M \to T_{c^k}M$ by $\psi_{c^k} = \exp_{c^k}^{-1} \circ \phi_{c^k}$. Since both exp and ϕ are retractions,

$$\psi_{c^k}(0_{c^k}) = 0_{c^k},$$
$$D\psi_{c^k}|_{0_{c^k}} = \mathrm{id}_{T_{c^k}M}.$$

Thus, per definition of Fréchet derivatives,

$$\psi_{c^k}(x) - \psi_{c^k}(0_{c^k}) - D\psi_{c^k}|_{0_{c^k}} x = \psi_{c^k}(x) - x = o(x),$$

in particular: choosing $x = \phi_{c^k}^{-1}(u^{k+1})$ we get

$$\exp_{c^k}^{-1}(u^{k+1}) - \phi_{c^k}^{-1}(u^{k+1}) = o(\|\phi_{c^k}^{-1}(u^{k+1})\|_{c^k}),$$

meaning

$$\| \exp_{c^k}^{-1}(u^{k+1}) \|_{c^k} \le \|\phi_{c^k}^{-1}(u^{k+1})\|_{c^k} + o(\|\phi_{c^k}^{-1}(u^{k+1})\|_{c^k}). \tag{7.3.6}$$

Taking $c^k = u^k$ and combining (7.3.5) and (7.3.6) we find

$$d(u^k, u^{k+1}) = \| \exp_{c^k}^{-1}(u^{k+1}) \|_{c^k} \le \|\phi_{c^k}^{-1}(u^{k+1})\|_{c^k} + o(\|\phi_{c^k}^{-1}(u^{k+1})\|_{c^k}).$$

Hence, since $\left\| \phi_{c^k}^{-1}(u^k) - \phi_{c^k}^{-1}(u^{k+1}) \right\|_{c^k} = \left\| \phi_{c^k}^{-1}(u^{k+1}) \right\|_{c^k}$ when $c^k = u^k$,

$$\lim_{k\to\infty} d(u^k, u^{k+1}) \le \lim_{k\to\infty} \left\| \phi_{c^k}^{-1}(u^k) - \phi_{c^k}^{-1}(u^{k+1}) \right\|_{c^k} = 0. \tag{7.3.7}$$

Note that we can exchange the roles of u^k and u^{k+1} and obtain the same result.

Since V is bounded from below, the sublevel sets M_μ of V are the preimages of the closed subsets $[C, \mu]$ and are hence closed as well. Since V is assumed to be coercive, by Theorem 7.1 the M_μ are bounded, and so since M is geodesically complete, by the Hopf-Rinow theorem the M_μ are compact [26]. In particular, $M_{V(u^0)}$ is compact such that $\overline{\operatorname{grad}} V$ is uniformly continuous on $M_{V(u^0)} \times M_{V(u^0)}$ by the Heine-Cantor theorem. This means that for any $\epsilon > 0$

there exists $\delta > 0$ such that if $d_{M\times M}((u^k, u^{k+1}), (u^k, u^k)) = d_M(u^k, u^{k+1}) < \delta$, then

$$\left\|\overline{\text{grad}}V(u^k, u^{k+1}) - \text{grad}V(u^k)\right\|_{c^k} = \left\|\overline{\text{grad}}V(u^k, u^{k+1}) - \overline{\text{grad}}V(u^k, u^k)\right\|_{c^k} < \epsilon.$$

Since $d_M(u^k, u^{k+1}) \to 0$, given $\epsilon > 0$ there exists K such that for all $k > K$,

$$\left\|\text{grad}V(u^k)\right\|_{c^k} \leq \left\|\overline{\text{grad}}V(u^k,u^{k+1}) - \text{grad}V(u^k)\right\|_{c^k} + \left\|\overline{\text{grad}}V(u^k,u^{k+1})\right\|_{c^k} \leq 2\epsilon.$$

This means

$$\lim_{k\to\infty} \text{grad}V(u^k) = 0.$$

Since $M_{V(u^0)}$ is compact, there exists a convergent subsequence $\{u^{k_l}\}$ with limit u^*. Since V is continuously differentiable,

$$\text{grad}V(u^*) = \lim_{l\to\infty} \text{grad}V(u^{k_l}) = 0 \quad \square$$

Remark: In the above proof, we assumed $c^k = u^k$ or $c^k = u^{k+1}$. Although these choices may be desirable for practical purposes, as discussed in the next subsection, one can also make a more general choice. Specifically, if $\phi = \exp$ and c^k, let $\gamma^k(t)$ be the geodesic between u^k and u^{k+1} such that

$$\gamma^k(t) = \exp_{u^k}(t v^k)$$

where $v^k = \exp_{u^k}^{-1}(u^{k+1})$. Then, taking $c^k = \gamma^k(s)$ for some $s \in [0,1]$, uniqueness of geodesics implies that

$$\exp_{c^k}(t\dot\gamma^k(s)) = \exp_{u^k}((t+s)v^k).$$

Hence,

$$\exp_{c^k}^{-1}(u^k) = -s\dot\gamma^k(s), \qquad \exp_{c^k}^{-1}(u^{k+1}) = (1-s)\dot\gamma^k(s),$$

and so, since geodesics are constant speed curves:

$$d(u^k, u^{k+1}) = \|v\|_{u^k} = \|\dot\gamma^k(s)\|_{c^k} = \|\exp_{c^k}^{-1}(u^k) - \exp_{c^k}^{-1}(u^{k+1})\|_{c^k}.$$

This means that (7.3.7) holds in this case. No other arguments in Theorem 7.2 are affected.

7.3.1 Itoh–Abe discrete Riemannian gradient

The Itoh–Abe discrete gradient [16] can be generalized to Riemannian manifolds.

Proposition 7.2. *Given a continuously differentiable energy $V : M \to \mathbb{R}$ and an orthogonal basis $\{E_j\}_{j=1}^n$ for $T_{c(u,v)}M$ such that*

$$\phi_c^{-1}(v) - \phi_c^{-1}(u) = \sum_{i=1}^{n} \alpha_i E_i,$$

define $\overline{\operatorname{grad}}_{\mathrm{IA}} V : M \times M \to T_{c(u,v)}M$ by

$$\overline{\operatorname{grad}}_{\mathrm{IA}} V(u,v) = \sum_{j=1}^{n} a_j E_j,$$

where

$$a_j = \begin{cases} \dfrac{V(w_j) - V(w_{j-1})}{\alpha_j}, & \alpha_j \neq 0 \\ g(\operatorname{grad} V(w_{j-1}), d\phi_c|_{\eta_{j-1}} E_j), & \alpha_j = 0. \end{cases}$$

$$w_j = \phi_c(\eta_j), \quad \eta_j = \phi_c^{-1}(u) + \sum_{i=1}^{j} \alpha_i E_i.$$

Then, $\overline{\operatorname{grad}}_{\mathrm{IA}} V$ is a discrete Riemannian gradient.

Proof. Continuity of $\overline{\operatorname{grad}}_{\mathrm{IA}} V$ can be seen from the smoothness of the local coordinate frame $\{E_j\}_{j=1}^n$ and from the continuity of the $a_j(\alpha_j)$:

$$\begin{aligned} \lim_{\alpha_j \to 0} a_j(\alpha_j) &= \lim_{\alpha_j \to 0} \frac{V\left(\phi_c\left(\eta_{j-1} + \alpha_j E_j\right)\right) - V\left(\phi_c\left(\eta_{j-1}\right)\right)}{\alpha_j} \\ &= \frac{\mathrm{d}}{\mathrm{d}\alpha_j}\bigg|_{\alpha_j = 0} V\left(\phi_c\left(\eta_{j-1} + \alpha_j E_j\right)\right) \\ &= \left\langle \mathrm{d}V\left(\phi_c\left(\eta_{j-1}\right)\right), d\phi_c|_{\eta_{j-1}} E_j \right\rangle \\ &= g(\operatorname{grad} V(w_{j-1}), d\phi_c|_{\eta_{j-1}} E_j). \end{aligned}$$

Property (7.3.1) holds since

$$\begin{aligned} g\left(\overline{\operatorname{grad}}_{\mathrm{IA}} V(u,v), \phi_c^{-1}(v) - \phi_c^{-1}(u)\right) &= \sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i a_j g(E_i, E_j) \\ &= \sum_{j=1}^{n} V(w_j) - V(w_{j-1}) \\ &= V(w_n) - V(w_0) \\ &= V(v) - V(u). \end{aligned}$$

Furthermore, (7.3.2) holds since when $v = u$, all $\alpha_j = 0$ and $c(u,v) = u$ so that

$$\overline{\operatorname{grad}}_{\mathrm{IA}} V(u,u) = \sum_{j=1}^{n} g(\operatorname{grad} V(u), E_j) E_j = \operatorname{grad} V(u). \quad \square$$

The map $\overline{\operatorname{grad}}_{\mathrm{IA}} V$ is called the Itoh–Abe discrete Riemannian gradient. For the Itoh–Abe DRG to be a computationally viable option it is important to compute the α_i efficiently. Consider for instance the gradient flow system. Applying the Itoh–Abe DRG to this we get the scheme

$$\begin{aligned} u^{k+1} &= \phi_{c^k}\left(W(u^k, u^{k+1})\right), \\ W(u^k, u^{k+1}) &= \phi_{c^k}^{-1}(u^k) - \tau_k \overline{\operatorname{grad}}_{\mathrm{IA}} V(u^k, u^{k+1}), \end{aligned}$$

meaning

$$\phi_{c^k}^{-1}(u^{k+1}) - \phi_{c^k}^{-1}(u^k) = -\tau_k \overline{\operatorname{grad}}_{\mathrm{IA}} V(u^k, u^{k+1}),$$

and in coordinates

$$\sum_{i=1}^{n} \alpha_i E_i = -\tau_k \sum_{j=1}^{n} \frac{V(w_j) - V(w_{j-1})}{\alpha_j} E_j,$$

so that the α_i are found by solving the n coupled equations

$$\alpha_i = -\tau_k \frac{V(w_i) - V(w_{i-1})}{\alpha_i}.$$

Note that these equations in general are fully implicit in the sense that they require knowledge of the endpoint u^{k+1} since the w_i are dependent on c^k. However, if we take $c^k = u^k$, there is no dependency on the endpoint and all the above equations become scalar, although one must solve them successively. For this choice of c^k we present, as Algorithm 1, a procedure for solving the gradient flow problem on a Riemannian manifold with Riemannian metric g using the Itoh–Abe DRG.

Algorithm 7.1. DRG-OPTIM

Choose $tol > 0$ and $u^0 \in M$. Set $k = 0$.
repeat
  Choose τ_k and an orthogonal basis $\{E_i^k\}_{i=1}^n$ for $T_{u^k}M$
  $v_0^k = u^k$
  $w_0^k = \phi_{u^k}^{-1}(v_0^k)$
  for $j = 1, ..., n$ **do**
    Solve $\alpha_j^k = -\tau_k \left(V\left(\phi_{u^k}(w_{j-1}^k + \alpha_j^k E_j^k)\right) - V\left(v_{j-1}^k\right)\right) / \alpha_j^k$
    $w_j^k = w_{j-1}^k + \alpha_j^k E_j^k$
    $v_j^k = \phi_{u^k}(w_j^k)$
  end for
  $u^{k+1} = v_n^k$
  $k = k + 1$
until $\left(V(u^k) - V(u^{k-1})\right) / V(u^0) < tol$

There is a caveat to this algorithm in that the α_j^k should be easy to compute. For example, it is important that the E_j and ϕ are chosen such that the difference $V(\phi_{u^k}(w_{j-1}^k + \alpha_j^k E_j^k)) - V(v_{j-1}^k)$ is cheap to evaluate. One can use any equation solver in computing α_j^k. To stay in line with the derivative-free nature of Algorithm 1, one may wish to use a solver like the Brent–Dekker algorithm [3]. Also worth noting is that the parallelization procedure used in [20] works for Algorithm 1 as well.

7.4 Numerical experiments

This section concerns four applications of DRG methods to gradient flow systems. In each case, we specify all details needed to implement Algorithm 1: the manifold M, retraction ϕ, and basis vectors $\{E_k\}$. The first two examples are eigenvalue problems, included to illuminate implementational issues with examples in a familiar setting. We do not claim that our algorithm is competitive with other eigenvalue solvers, but include these examples for the sake of exposition and to have problems with readily available reference solutions. The first of these is a simple Rayleigh quotient minimization problem, where issues of computational efficiency are raised. The second one concerns the Brockett flow on $\mathrm{SO}(m)$, the space of orthogonal $m \times m$ matrices with unit determinant, and serves as an example of optimization on a Lie group. The remaining two problems are examples of manifold-valued image analysis problems concerning Interferometric Synthetic Aperture Radar (InSAR) imaging and Diffusion Tensor Imaging (DTI), respectively. Specifically, the problems concern total variation denoising of images obtained through these techniques [28]. The

experiments do not consider the quality of the solution paths, i.e. numerical accuracy. For experiments of this kind, we refer to [5].

All programs used in the following were implemented as MATLAB functions, with critical functions implemented in C using the MATLAB EXecutable (MEX) interface when necessary. The code was executed using MATLAB (2017a release) running on a Mid 2014 MacBook Pro with a four-core 2.5 GHz Intel Core i7 processor and 16 GB of 1600 MHz DDR3 RAM. We used a C language port of the built-in MATLAB function `fzero` for the Brent-Dekker algorithm implementation.

7.4.1 Eigenvalue problems

As an expository example, our first problem consists of finding the smallest eigenvalue/vector pair of a symmetric $m \times m$ matrix A by minimizing its Rayleigh quotient. We shall solve this problem using both the extrinsic and intrinsic view of the $(m-1)$-sphere. In the second example we consider the different approach to the eigenvalue problem proposed by Brockett in [4]. Here, the gradient flow on $\mathrm{SO}(m)$ produces a diagonalizing matrix for a given symmetric matrix.

Eigenvalues via Rayleigh quotient minimization

In our first example, we wish to compute the smallest eigenvalue of a symmetric matrix $A \in \mathbb{R}^{m\times m}$ by minimizing the Rayleigh quotient

$$V(u) = u^T A u$$

with u on the $(m-1)$-sphere S^{m-1}.

Taking the extrinsic view, we regard S^{m-1} as a submanifold of $\mathbb{R}^m$, equipped with the standard Euclidian metric $g(x,y) = x^{\mathrm{T}}y$. In this representation, $T_u S^{m-1}$ is the hyperplane tangent to u, i.e. $T_u S^{m-1} = \{x \in \mathbb{R}^m : x^{\mathrm{T}}u = 0\}$. A natural choice of retraction is

$$\phi_p(x) = \frac{p+x}{\|p+x\|}.$$

There is a difficulty with this ϕ; it does not preserve sparsity, meaning Algorithm 1 will be inefficient as discussed above. To see this, consider that at each time step, to find the α_j^k, we must compute the difference

$$V(z_j^k) - V(z_{j-1}^k) = (z_j^k)^T A z_j^k - (z_{j-1}^k)^T A z_{j-1}^k$$

for some $z_{j-1}^k, z_j^k \in S^{m-1}$. We can compute this efficiently if $z_j^k = z_{j-1}^k + \delta$, where δ is sparse. Then,

$$V(z_j^k) - V(z_{j-1}^k) = 2(z_{j-1}^k)^T A\delta + \delta^T A\delta,$$

which is efficient since one may assume Az_{j-1}^k to be precomputed so that the computational cost is limited by the sparsity of δ. In our case, we have

$$z_{j-1}^k = \phi_c(w_{j-1}^k), \qquad z_j^k = \phi_c(w_{j-1}^k + \alpha_j^k E_j).$$

However, with ϕ_c as above, $\delta = \phi_c(w_{j-1}^k + \alpha_j^k E_j) - \phi_c(w_{j-1}^k)$ is non-sparse, and so computing the energy difference is costly.

Next, let us consider the intrinsic view of S^{m-1}, representing it in spherical coordinates $\theta \in \mathbb{R}^{m-1}$ by

$$\begin{aligned} u_1(\theta) &= \cos(\theta_1), \\ u_r(\theta) &= \cos(\theta_r) \prod_{i=1}^{r-1} \sin(\theta_i), \quad 1 < r < m, \\ u_m(\theta) &= \prod_{i=1}^{m-1} \sin(\theta_i). \end{aligned}$$

Due to the simple structure of $\mathbb{R}^{m-1}$, we take $\phi_\theta(\eta) = \theta + \eta$. Then, we have

$$u_r(\phi_\theta(\alpha E_l)) = u_r(\theta + \alpha E_l) = \begin{cases} u_r(\theta), & r < l \\ \dfrac{\cos(\theta_l + \alpha)}{\cos(\theta_l)} u_r(\theta), & r = l \\ \dfrac{\sin(\theta_l + \alpha)}{\sin(\theta_l)} u_r(\theta), & r > l. \end{cases}$$

Using this relation, the energy difference after a coordinate update becomes:

$$\begin{aligned} V(u(\theta+\alpha E_l)) - V(u(\theta)) = {} & 2\kappa_{1l} \sum_{i=1}^{l-1} u_i(\theta) u_l(\theta) A_{il} + 2\kappa_{2l} \sum_{i=1}^{l-1} \sum_{j=l+1}^{m} u_i(\theta) u_j(\theta) A_{ij} \\ & + 2\kappa_{3l} \sum_{j=l+1}^{m} u_l(\theta) u_j(\theta) A_{lj} + \kappa_{4l} \sum_{i=l+1}^{m} \sum_{j=l+1}^{m} u_i(\theta) u_j(\theta) A_{ij}. \\ & + \kappa_{5l} u_l(\theta) u_l(\theta) A_{ll}, \end{aligned}$$

with

$$\kappa_{1l} = c_l - 1, \quad \kappa_{2l} = s_l - 1, \quad \kappa_{3l} = s_l c_l - 1, \quad \kappa_{4l} = s_l^2 - 1, \quad \kappa_{5l} = c_l^2 - 1,$$

where

$$c_l = \frac{\cos(\theta_l + \alpha)}{\cos(\theta_l)}, \quad s_l = \frac{\sin(\theta_l + \alpha)}{\sin(\theta_l)}.$$

With prior knowledge of $V(u(\theta))$ (and thus the four partial sums in the difference), evaluating $V(u(\theta+\alpha E_l)) - V(u(\theta))$ amounts to five scalar multiplications

and four scalar additions after evaluating the κ_i^l. With correct bookkeeping, new sums can be evaluated from previous sums after coordinate updates, reducing the computational complexity of the algorithm. Although not producing an algorithm competitive with standard eigenvalue solvers, this example demonstrates that the correct choice of coordinates is vital to reducing the computational complexity of the Itoh–Abe DRG method.

Eigenvalues via Brockett flow

Among other things, the article of Brockett [4] discusses how one may find the eigenvalues of a symmetric matrix A by solving the following gradient flow problem on $M = \mathrm{SO}(m)$:

$$\dot{Q} = -Q(DQ^T AQ - Q^T AQD) \tag{7.4.1}$$

Here, D is a real diagonal matrix with non-repeated entries. It can be shown that $\lim_{t\to\infty} Q = Q^*$, where $(Q^*)^T AQ^* = \Lambda$ is diagonal and hence contains the eigenvalues of A, ordered as the entries of D. Equation (7.4.1) is the gradient flow of the energy

$$V(Q) = \mathrm{tr}(AQ^T DQ) \tag{7.4.2}$$

with respect to the trace metric on $\mathrm{SO}(m)$. One can check that $\mathrm{SO}(m)$ is a Lie group [27], with Lie algebra

$$\mathfrak{so}(m) = \{B \in \mathbb{R}^{m\times m} : B^T = -B\}.$$

Also, since $\mathrm{SO}(m)$ is a matrix Lie group, the exponential coincides with the matrix exponential. However, we may consider using some other function as a retraction, such as the Cayley transform $\phi : \mathfrak{so}(m) \to \mathrm{SO}(m)$ given by

$$\phi(B) = (I - B)^{-1}(I + B).$$

Figure 7.1 shows the results of numerical tests with constant time step $\tau_k = 0.1$ and $m = 20$. In the left hand panel, the evolution of the diagonal values of $Q^k AQ^k$ compared to the spectrum of A is shown; it is apparent that the diagonal values converge to the eigenvalues. The right hand panel shows the convergence rate of Algorithm 1 to the minimal value V^* as computed with eigenvalues and eigenvectors from MATLAB's `eigen` function. It would appear that the convergence rate is linear, meaning $\|D - (Q^{k+1})^T AQ^{k+1}\| = C\|D - (Q^k)^T AQ^k\|$, with $C < 1$, which corresponds to an exponential reduction in $\|D - (Q^k)^T AQ^k\|$. No noteworthy difference was observed when using the matrix exponential in place of the Cayley transform.

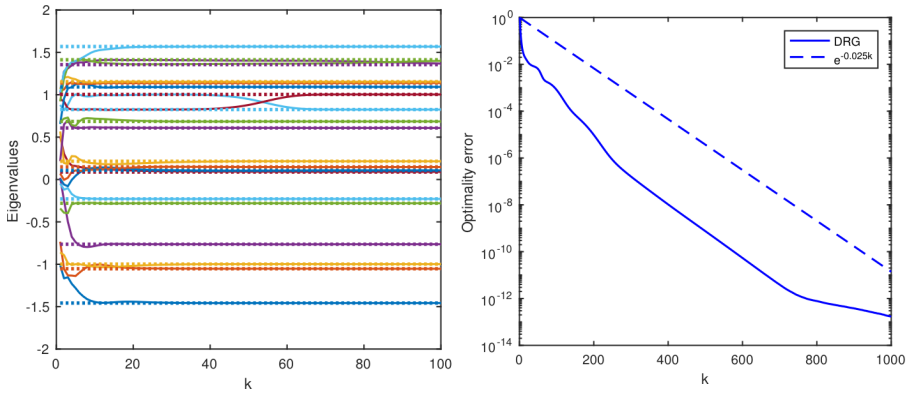


Figure 7.1: Brockett flow with $\tau_k = 0.1$ and 20 eigenvalues. Random initial matrix. Left: Evolution of eigenvalues. Right: Optimality error $(V(u^k) - V^*)/(V(u^0) - V^*)$.

7.4.2 Manifold valued imaging

In the following two examples we will consider problems from manifold valued 2D imaging. We will in both cases work on a product manifold $\mathcal{M} = M^{l\times m}$ consisting of $l \times m$ copies of an underlying data manifold M. An element of M will in this case be called an *atom*, as opposed to the regular term *pixel*. As explained in [18], product manifolds of Riemannian manifolds are again Riemannian manifolds. The tangent spaces of product manifolds have a natural structure as direct sums, with $T_{(u_{11},u_{12},\dots,u_{lm})}\mathcal{M} = \bigoplus_{i,j=1}^{l,m} T_{u_{ij}}M$, which induces a natural Riemannian metric $\mathcal{G} : T\mathcal{M} \times T\mathcal{M} \to \mathbb{R}$ fiberwise as

$$\mathcal{G}_{(u_{11},u_{12},\dots,u_{lm})}((x_{11},\dots,x_{lm}),(y_{11},\dots,y_{lm})) = \sum_{i,j=1}^{l,m} g_{u_{ij}}(x_{ij},y_{ij}).$$

Also, given a retraction $\phi : TM \to M$, one can define a retraction $\Phi : T\mathcal{M} \to \mathcal{M}$ fiberwise as

$$\Phi_{(u_{11},u_{12},\dots,u_{lm})}(x_{11},\dots,x_{lm}) = (\phi_{u_{11}}(x_{11}),\phi_{u_{12}}(x_{12}),\dots,\phi_{u_{lm}}(x_{lm})).$$

Discrete gradients were first used in optimization algorithms for image analysis in [10] and [20]. As an example of a manifold-valued imaging problem, consider Total Variation (TV) denoising of manifold valued images [28], where one wishes to minimize, based on generalizations of the L^β and L^γ norms:

$$V(u) = \frac{1}{\beta}\sum_{i,j=1}^{l,m} \mathrm{d}(u_{ij},s_{ij})^\beta + \lambda\left(\sum_{i,j=1}^{l-1,m} \mathrm{d}(u_{ij},u_{i+1,j})^\gamma + \sum_{i,j=1}^{l,m-1} \mathrm{d}(u_{ij},u_{i,j+1})^\gamma\right). \tag{7.4.3}$$

Here, $s = (s_{11}, ..., s_{lm}) \in \mathcal{M}$ is the input image, $u = (u_{11}, ..., u_{lm}) \in \mathcal{M}$ is the output image, λ is a regularization strength constant, and d is a metric on M, which we will take to be the geodesic distance induced by g.

InSAR image denoising

We first consider Interferometric Synthetic Aperture Radar (InSAR) imaging, used in earth observation and terrain modelling [22]. In InSAR imaging, terrain elevation is measured by means of phase differences between laser pulses reflected from a surface at different times. Thus, the atoms g_{ij} are elements of $M = S^1$, represented by their phase angles: $-\pi < g_{ij} \leq \pi$. After processing, the phase data is *unwrapped* to form a single, continuous image of displacement data [8]. The natural distance function in this representation is the angular distance

$$\mathrm{d}(\varphi, \theta) = \begin{cases} |\varphi - \theta|, & |\varphi - \theta| \leq \pi \\ 2\pi - |\varphi - \theta|, & |\varphi - \theta| > \pi. \end{cases}$$

Also, $T_\varphi M$ is simply $\mathbb{R}$, and ϕ is given, with $\underset{2\pi}{+}$ denoting addition modulo 2π, as:

$$\phi_\varphi(\theta_\varphi) = (\theta \underset{2\pi}{+} (\varphi + \pi)) - \pi.$$

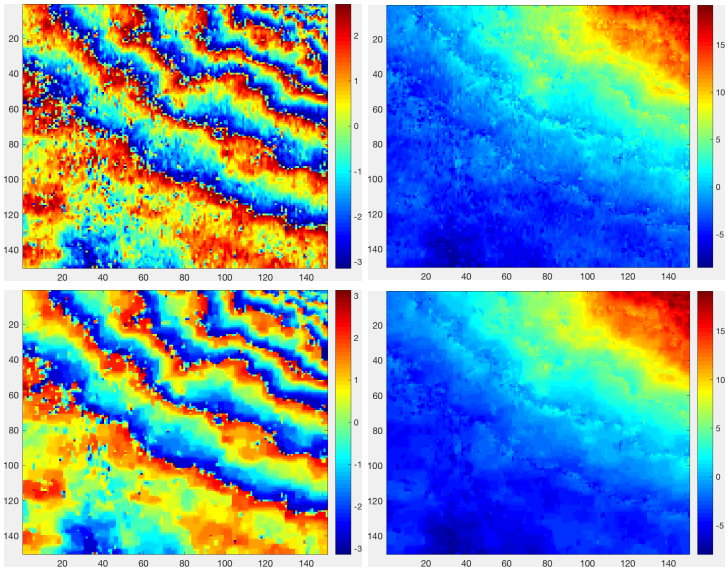


Figure 7.2: Left column: Interferogram. Right column: Phase unwrapped image. Top row: Original image. Bottom row: Denoising with $\beta = 2$, $\lambda = 0.3$.

Figure 7.2 shows the result of applying TV denoising to an InSAR image of a slope of Mt. Vesuvius, Italy, with $\beta = 2$. The left column shows the phase data, while the right hand side shows the phase unwrapped data. The input image was taken from [21]. It is evident that the algorithm is successful in removing noise. Computation time was 0.1 seconds per iteration on a 150×150 image.

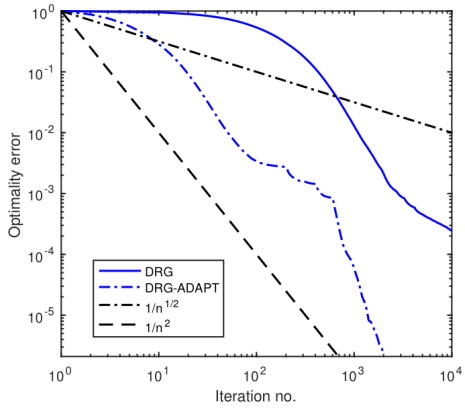


Figure 7.3: Logarithmic plot of optimality error $(V(u^k) - V^*)/(V(u^0) - V^*)$.

A logarithmic plot showing convergence in terms of $(V(u^k) - V^*)/(V(u^0) - V^*)$ is shown in Figure 7.3, where V^* is a near-optimal value for V, obtained by iterating until $V(u^{k+1}) - V(u^k) \leq 10^{-15}$. The plot shows the behaviour of Algorithm 7.1 with constant time steps $\tau_k = \tau_0 = 0.002$ and an ad-hoc adaptive method with $\tau_0 = 0.005$ where τ_k is halved each 200 iterations; for each of these strategies a separate V^* was found since they did not produce convergence to the same minimizer. The reason for the different minimizers is that the TV functional, and thus the minimization problem, is non-convex in S^1 [25]. We can observe that the convergence speed varies between $\mathcal{O}(1/k)$ and $\mathcal{O}(1/k^2)$, with faster convergence for the ad-hoc adaptive method. The reason for this sublinear convergence as compared to the linear convergence observed in the Brockett flow case may be the non-convexity.

DTI image denoising

Diffusion Tensor Imaging (DTI) is a medical imaging technique where the goal is to make spatial samples of the tensor specifying the diffusion rates of water in biological tissue. The tensor is assumed to be, at each point (i, j), represented by a matrix $A_{ij} \in \mathrm{Sym}^+(3)$, the space of 3×3 symmetric positive definite (SPD) matrices. Experimental measurements of DTI data are, as with other MRI techniques, contaminated by Rician noise [11], which one may attempt to remove

by minimizing (7.4.3) with an appropriate choice of Riemannian structure on $\mathcal{M} = \mathrm{Sym}^+(3)^{m\times l}$.

As above, since the manifold we are working on is a product manifold, it suffices to define the Riemannian structure on $\mathrm{Sym}^+(3)$. First off, one should note that $T_A\mathrm{Sym}^+(3)$ can be identified with $\mathrm{Sym}(3)$, the space of symmetric 3×3 matrices [17]. In [28], the authors consider equipping $\mathrm{Sym}^+(3)$ with the affine invariant Riemannian metric given pointwise as

$$g_A(X,Y) = \mathrm{tr}(A^{-\frac{1}{2}}XA^{-1}YA^{-\frac{1}{2}}),$$

and for purposes of comparison, so shall we. The space $\mathrm{Sym}^+(3)$ equipped with this metric is a Cartan-Hadamard manifold [17], and thus is complete, meaning that Theorem 7.2 holds. This metric induces the explicitly computable geodesic distance

$$d(A,B) = \sqrt{\sum_{i=1}^{3}\log(\kappa_i)^2}$$

on $\mathrm{Sym}^+(3)$, where κ_i are the eigenvalues of $A^{-\frac{1}{2}}BA^{-\frac{1}{2}}$. Furthermore, the metric induces a Riemannian exponential given by

$$\exp_A(Y) = A^{1/2}\mathrm{e}^{A^{-1/2}YA^{-1/2}}A^{1/2}$$

where e denotes the matrix exponential, and $A^{1/2}$ is the matrix square root of A. We could choose the retraction as $\phi = \exp$, but there are less computationally expensive options that do not involve computing matrix exponentials. More specifically, we will make use of the second-order approximation of the exponential,

$$\phi_A(Y) = A + Y + \frac{1}{2}YA^{-1}Y.$$

While a first-order expansion is also a retraction, there is no guarantee that $A + Y \in \mathrm{Sym}^+(3)$, whereas the second-order expansion, which can be written on the form

$$\phi_A(Y) = \frac{1}{2}A + \frac{1}{2}(A^{\frac{1}{2}} + A^{-\frac{1}{2}}Y)^T(A^{\frac{1}{2}} + A^{-\frac{1}{2}}Y),$$

is clearly symmetric positive definite since A is so. Note that using a sparse basis E_{ij} (in our example we use $E_{ij} = e_ie_j^T + e_je_i^T$) for the space $\mathrm{Sym}(3)$, evaluating $\phi_A(X + \alpha E_{ij})$ amounts to, at most, four scalar updates when $\phi_A(X)$ and A^{-1} is known, as is possible with proper bookkeeping in the software implementation. Also, since all matrices involved are 3×3 SPD matrices, one may

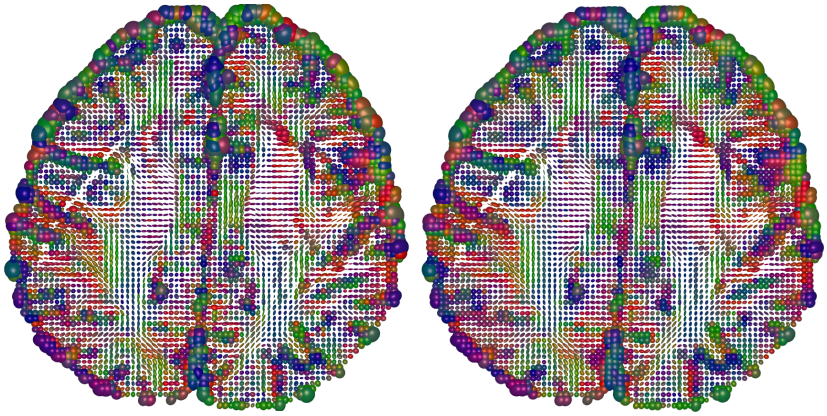


Figure 7.4: DTI scan, axial slice. Left: Noisy image. Right: Denoised with $\beta = 2$, $\lambda = 0.05$.

find eigenvalues and eigenvectors directly, thus allowing for fast computations of matrix square roots and, consequently, geodesic distances.

Figure 7.4 shows an example of denoising DTI images using the TV regularizer. The data is taken from the publicly available Camino data set [7]. The DTI tensor has been calculated from underlying data using linear least-squares fitting, and is subject to Rician noise (left hand side), which is mitigated by TV denoising (right hand side). The denoising procedure took about 7 seconds for 57 iterations, on a 72×73 image. The algorithm was stopped when the relative change in energy, $(V(u^0) - V(u^k))/V(u^0)$ dropped below 10^{-5}. Each atom $A \in \mathrm{Sym}^+(3)$ is visualized by an ellipsoid with the eigenvectors of A as principal semi-axes, scaled by the corresponding eigenvalues. The colors are coded to correspond to the principal direction of the major axis, with red denoting left-right orientation, green anterior-posterior and blue inferior-superior.

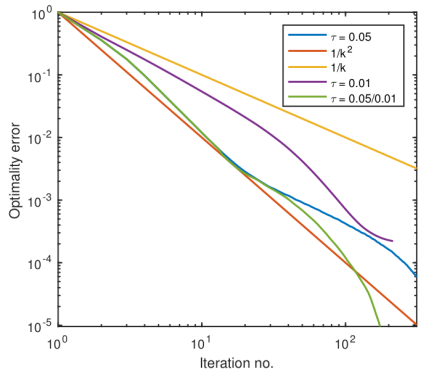


Figure 7.5: Logarithmic plot of optimality error.

Figure 7.5 shows the convergence behaviour of Algorithm 1, with three different time steps: $\tau = 0.05$, $\tau = 0.01$ and a mixed strategy of using $\tau = 0.05$ for 12 steps, then changing to $\tau = 0.01$. Also, baseline rates of $1/k^2$ and $1/k$ are shown. It is apparent that the choice of time step has great impact on the convergence rate, and that simply changing the time step from $\tau = 0.05$ to $\tau = 0.01$ is effective in speeding up convergence. This would suggest that time step adaptivity is a promising route for acceleration of these methods.

7.5 Conclusion and future work

We have extended discrete gradient methods to Riemannian manifolds, and shown how they may be applied to gradient flows. The Itoh–Abe discrete gradient has been formulated in a manifold setting; this is, to the best of our knowledge, the first time this has been done. In particular, we have used the Itoh–Abe DRG on gradient systems to produce a derivative-free optimization algorithm on Riemannian manifolds. This optimization algorithm has been proven to converge under reasonable conditions, and shows promise when applied to the problem of denoising manifold valued images using the total variation approach of [28].

As with the algorithm in the Euclidian case, there are open questions. The first question is which convergence rate estimates can be made; one should especially consider the linear convergence exhibited in the Brockett flow problem, and the rate observed in Figure 7.5 which approaches $1/k^2$. A second question is how to formulate a rule for choosing step sizes so as to accelerate convergence toward minimizers. There is also the question of how the DRG methods perform as ODE solvers for dissipative problems on Riemannian manifolds; in particular, convergence properties, stability, and convergence order. The above discussion is geared toward optimization applications due to the availability of optimization problems, but it would be of interest to see how the methods work as ODE solvers in their own right similar to the analysis and experiments done in [5].

Bibliography

[1] P.-A. Absil, R. Mahony, and R. Sepulchre, *Optimization algorithms on matrix manifolds*, Princeton University Press, 2009.

[2] R. L. Adler, J.-P. Dedieu, J. Y. Margulies, M. Martens, and M. Shub, *Newton's method on Riemannian manifolds and a geometric model for the human spine*, IMA J. Numer. Anal., 22 (2002), pp. 359–390.

[3] R. P. BRENT, *An algorithm with guaranteed convergence for finding a zero of a function*, Comput. J., 14 (1971), pp. 422–425.

[4] R. W. BROCKETT, *Dynamical systems that sort lists, diagonalize matrices and solve linear programming problems*, in IEEE Decis. Contr. P., IEEE, 1988, pp. 799–803.

[5] E. CELLEDONI, S. EIDNES, B. OWREN, AND T. RINGHOLM, *Energy preserving methods on Riemannian manifolds*, (2018).

[6] E. CELLEDONI AND B. OWREN, *Preserving first integrals with symmetric Lie group methods*, Discrete Cont. Dyn. S., 34 (2014), pp. 977–990.

[7] P. COOK, Y. BAI, S. NEDJATI-GILANI, K. SEUNARINE, M. HALL, G. PARKER, AND D. ALEXANDER, *Camino: open-source diffusion-MRI reconstruction and processing*, in Proc. 14th Sci. Meeting of ISMRM, vol. 2759, Seattle WA, USA, 2006.

[8] R. M. GOLDSTEIN, H. A. ZEBKER, AND C. L. WERNER, *Satellite radar interferometry: Two-dimensional phase unwrapping*, Radio Sci., 23 (1988), pp. 713–720.

[9] O. GONZALEZ, *Time integration and discrete Hamiltonian systems*, J. Nonlinear Sci., 6 (1996), pp. 449–467.

[10] V. GRIMM, R. I. MCLACHLAN, D. I. MCLAREN, G. QUISPEL, AND C. SCHÖNLIEB, *Discrete gradient methods for solving variational image regularisation models*, J. Phys. A: Math. Theor., 50 (2017), p. 295201.

[11] H. GUDBJARTSSON AND S. PATZ, *The Rician distribution of noisy MRI data*, Magn. Reson. Med., 34 (1995), pp. 910–914.

[12] E. HAIRER AND C. LUBICH, *Energy-diminishing integration of gradient systems*, IMA J. Numer. Anal., 34 (2013), pp. 452–461.

[13] E. HAIRER, C. LUBICH, AND G. WANNER, *Geometric numerical integration*, vol. 31 of Springer Series in Computational Mathematics, Springer, Heidelberg, 2010.

[14] A. HARTEN, P. D. LAX, AND B. VAN LEER, *On upstream differencing and Godunov-type schemes for hyperbolic conservation laws*, SIAM Rev., 25 (1983), pp. 35–61.

[15] A. HUMPHRIES AND A. STUART, *Runge–Kutta methods for dissipative and gradient dynamical systems*, SIAM J. Numer. Anal., 31 (1994), pp. 1452–1485.

[16] T. ITOH AND K. ABE, *Hamiltonian-conserving discrete canonical equations based on variational difference quotients*, J. Comput. Phys., 76 (1988), pp. 85–102.

[17] S. LANG, *Fundamentals of differential geometry*, vol. 191, Springer Science & Business Media, 2012.

[18] J. M. LEE, *Riemannian manifolds: an introduction to curvature*, vol. 176, Springer Science & Business Media, 2006.

[19] R. I. MCLACHLAN, G. R. W. QUISPEL, AND N. ROBIDOUX, *Geometric integration using discrete gradients*, Philos. T. R. Soc. A, 357 (1999), pp. 1021–1045.

[20] T. RINGHOLM, J. LAZIĆ, AND C.-B. SCHÖNLIEB, *Variational image regularization with Euler's elastica using a discrete gradient scheme*, arXiv preprint arXiv:1712.07386, (2017).

[21] F. ROCCA, C. PRATI, AND A. FERRETTI, *An overview of ERS-SAR interferometry*, in ERS Symp. Space Serv. Env., 1997.

[22] P. A. ROSEN, S. HENSLEY, I. R. JOUGHIN, F. K. LI, S. N. MADSEN, E. RODRIGUEZ, AND R. M. GOLDSTEIN, *Synthetic aperture radar interferometry*, P. IEEE, 88 (2000), pp. 333–382.

[23] S. SATO, T. MATSUO, H. SUZUKI, AND D. FURIHATA, *A Lyapunov-type theorem for dissipative numerical integrators with adaptive time-stepping*, SIAM J. Numer. Anal., 53 (2015), pp. 2505–2518.

[24] M. SHUB, *Some remarks on dynamical systems and numerical analysis*, P. VII ELAM., (1986), pp. 69–92.

[25] E. STREKALOVSKIY AND D. CREMERS, *Total variation for cyclic structures: Convex relaxation and efficient minimization*, in Proc. Cvpr. IEEE, IEEE Computer Society, 2011, pp. 1905–1911.

[26] C. UDRISTE, *Convex functions and optimization methods on Riemannian manifolds*, vol. 297, Springer Science & Business Media, 1994.

[27] F. W. WARNER, *Foundations of differentiable manifolds and Lie groups*, vol. 94, Springer Science & Business Media, 2013.

[28] A. WEINMANN, L. DEMARET, AND M. STORATH, *Total variation regularization for manifold-valued data*, SIAM J. Imaging Sci., 7 (2014), pp. 2226–2257.

Energy preserving methods on Riemannian manifolds

Elena Celledoni, Sølve Eidnes, Brynjulf Owren and Torbjørn Ringholm

Submitted

Energy preserving methods on Riemannian manifolds

Abstract. The energy preserving discrete gradient methods are generalized to finite-dimensional Riemannian manifolds by definition of a discrete approximation to the Riemannian gradient, a retraction, and a coordinate center function. The resulting schemes are intrinsic and do not depend on a particular choice of coordinates, nor on embedding of the manifold in a Euclidean space. Generalizations of well-known discrete gradient methods, such as the average vector field method and the Itoh–Abe method are obtained. It is shown how methods of higher order can be constructed via a collocation-like approach. Local and global error bounds are derived in terms of the Riemannian distance function and the Levi-Civita connection. Some numerical results on spin system problems are presented.

8.1 Introduction

A first integral of an ordinary differential equation (ODE) is a scalar-valued function on the phase space of the ODE that is preserved along solutions. The potential benefit of using numerical methods that preserve one or more such invariants is well-documented, and several energy-preserving methods have been developed in recent years. Among these are the discrete gradient methods, which were introduced for use in Euclidean spaces in [9], see also [19]. These methods are based on the idea of expressing the ODE using a skew-symmetric operator and the gradient of the first integral, and then creating a discrete counterpart to this in such a way that the numerical scheme preserves the energy.

For manifolds in general, one can use the same schemes expressed in local coordinates. A drawback is that the numerical approximation will typically depend on the particular choice of coordinates and also on the strategy used for transition between coordinate charts. Another alternative is to use a global embedding of the manifold into a larger Euclidean space, but then it typically happens that the numerical solution deviates from the manifold. Even if the situation can be amended by using projection, it may not be desirable that the computed approximation depends on the particular embedding chosen. Crouch and Grossmann [7] and Munthe-Kaas [20, 21] introduced different ways of extending existing Runge–Kutta methods to a large class of differentiable manifolds. Both these approaches are generally classified as Lie group integrators, see [12] or the more recent [4] for a survey of this class of methods. They can also both be formulated abstractly by means of a post-Lie structure which consists of a Lie algebra with a flat connection of constant torsion, see e.g. [22]. In the present paper we shall state the methods in a slightly different context,

This work was supported by the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 691070.

using the notion of a Riemannian manifold. It is then natural to make use of the Levi-Civita connection, which in contrast to the post-Lie setting is torsion-free, and which in general has a non-zero curvature. For our purposes it is also an advantage that the Riemannian metric provides an intrinsic definition of the gradient. Taking an approach more in line with this, Leimkuhler and Patrick [15] considered mechanical systems on the cotangent bundle of a Riemannian manifold and succeeded in generalising the classical leap-frog scheme to a symplectic integrator on Riemannian manifolds.

Some classical numerical methods in Euclidean spaces preserve certain classes of invariants; for instance, symplectic Runge–Kutta methods preserve all quadratic invariants. This can be useful when there is a natural way of embedding a manifold into a linear space by using constraints that are expressed by means of such invariants. An example is the 2-sphere which can be embedded in $\mathbb{R}^3$ by adding the constraint that these vectors should have unit length. The classical midpoint rule will automatically ensure that the numerical approximations remain on the sphere as it preserves all quadratic invariants. In general, however, the invariants preserved by these methods are expressed in terms of coordinates. Hence the preservation property of the method may be lost under coordinate changes if the invariant is no longer quadratic. In [5], a generalization of the discrete gradient method to differential equations on Lie groups and a broad class of manifolds was presented. Here we develop this further by introducing a Riemannian structure that can be used to provide an intrinsic definition of the gradient as well as a means to measure numerical errors.

The structure of this paper is as follows: In section 2, we formulate the problem to be solved and introduce discrete Riemannian gradient methods, as well as presenting some particular examples with special attention to a generalization of the Itoh–Abe discrete gradient. We also briefly discuss the Euclidean setting as a special choice of manifold and show how the standard discrete gradient methods are recovered in this case. In the third section, we consider higher order energy preserving methods based on generalization of a collocation strategy introduced by Hairer [10] to Riemannian manifolds. We present some error analysis in section 4, and show numerical results in section 5, where the methods are applied to spin system problems.

8.2 Energy preservation on Riemannian manifolds

Consider an initial value problem on the finite-dimensional Riemannian manifold (M, g),

$$\dot{u} = F(u), \quad u(0) = u^0 \in M. \tag{8.2.1}$$

We denote by $\mathcal{F}(M)$ the space of smooth functions on M. The set of smooth vector fields and differential one-forms are denoted $\Gamma(TM)$ and $\Gamma(T^*M)$ respectively, and for the duality pairing between these two spaces we use the angle brackets $\langle\cdot,\cdot\rangle$.

A first integral associated to a vector field $F \in \Gamma(TM)$ is a function $H \in \mathcal{F}(M)$ such that $\langle \mathrm{d}H, F\rangle$ vanishes identically on M. First integrals are preserved along solutions of (8.2.1),

$$\frac{\mathrm{d}}{\mathrm{d}t} H(u(t)) = \big\langle \mathrm{d}H(u(t)), \dot{u}(t) \big\rangle = \big\langle \mathrm{d}H(u(t)), F(u(t)) \big\rangle = 0.$$

8.2.1 Preliminaries

The fact that a vector field F has a first integral H is closely related to the existence of a tensor field $\Omega \in \Gamma(TM \otimes T^*M) =: \Gamma(\mathcal{T}^1_1 M)$, skew-symmetric with respect to the metric g, such that

$$F(u) = \Omega(u)\,\mathrm{grad}H(u), \tag{8.2.2}$$

where $\mathrm{grad}H \in \Gamma(TM)$ is the Riemannian gradient, the unique vector field satisfying $\langle \mathrm{d}H, \cdot\rangle = g(\mathrm{grad}H, \cdot)$. Any ODE (8.2.1) where F is of this form preserves H, since

$$\frac{\mathrm{d}}{\mathrm{d}t} H(u) = \big\langle \mathrm{d}H(u), \dot{u}\big\rangle = \big\langle \mathrm{d}H(u), \Omega\,\mathrm{grad}H(u)\big\rangle = g(\mathrm{grad}H(u), \Omega\,\mathrm{grad}H(u)) = 0.$$

A converse result is detailed in the following proposition.

Proposition 8.1. *Any system (8.2.1) with a first integral H can be written with an F of the form (8.2.2). The skew tensor field Ω can be chosen so as to be bounded near every nondegenerate critical point of H.*

Proof. Similar to the proof of Proposition 2.1 in [19], we can write an explicit expression for a possible choice of Ω,

$$\Omega y = \frac{g(\mathrm{grad}H, y)\,F - g(F, y)\,\mathrm{grad}H}{g(\mathrm{grad}H, \mathrm{grad}H)}. \tag{8.2.3}$$

Clearly, $g(y, \Omega y) = 0$ for all y. Since H is a first integral, $g(F, \mathrm{grad}H) = \langle \mathrm{d}H, F\rangle = 0$, so $\Omega\,\mathrm{grad}H = F$. For a proof that Ω is bounded near nondegenerate critical points, see [19]. □

In fact, such a tensor field Ω often arises naturally from a two-form ω through $\Omega y = \omega(\cdot, y)^\sharp$. A well-known example is when ω is a symplectic two-form. Note that Ω is not necessarily unique.

Retractions, viewed as maps from TM to M, will play an important role in the methods we discuss here. Their formal definition can be found e.g. in [1]:

Definition 8.1. Let ϕ be a smooth map $\phi : TM \to M$ and let ϕ_p denote the restriction of ϕ to T_pM, with 0_p being the zero-vector in T_pM. Then ϕ is a *retraction* if it satisfies the conditions

1. ϕ_p is defined in an open ball $B_{r_p}(0_p) \subset T_pM$ of radius r_p about 0_p,
2. $\phi_p(x) = p$ if and only if $x = 0_p$,
3. $T\phi_p\big|_{0_p} = \mathrm{Id}_{T_pM}$.

A generic example of a retraction on (M, g) is obtained via the Riemannian exponential, setting $\phi_p(x) = \exp_p(x)$, i.e. following along the geodesic emanating from p in the direction x.

8.2.2 The discrete Riemannian gradient method

We adapt the discrete gradients in Euclidean space to discrete Riemannian gradients (DRG) on (M, g) by means of a retraction map ϕ and a center point function c.

Definition 8.2. A discrete Riemannian gradient is a triple $(\overline{\mathrm{grad}}, \phi, c)$[1] where

1. $c : M \times M \to M$ is a continuous map such that $c(u, u) = u$ for all $u \in M$,
2. $\overline{\mathrm{grad}} : \mathcal{F}(M) \to \Gamma(c^*TM)$,
3. $\phi : TM \to M$ is a retraction,

such that for all $H \in \mathcal{F}(M)$, $u \in M$, $v \in M$, $c = c(u, v) \in M$,

$$H(v) - H(u) = g(\overline{\mathrm{grad}}H(u, v), \phi_c^{-1}(v) - \phi_c^{-1}(u)), \tag{8.2.4}$$

$$\overline{\mathrm{grad}}H(u, u) = \mathrm{grad}H(u). \tag{8.2.5}$$

The DRG $\overline{\mathrm{grad}}H$ is a continuous section of the pullback bundle c^*TM, meaning that $\pi \circ \overline{\mathrm{grad}}H = c$, where $\pi : TM \to M$ is the natural projection. We also need to define an approximation to be used for the tensor field $\Omega \in \Gamma(\mathcal{T}_1^1 M)$. To this end we let $\overline{\Omega} \in \Gamma(c^*\mathcal{T}_1^1 M)$ be a continuous skew-symmetric tensor field such that

$$\overline{\Omega}(u, u) = \Omega(u) \quad \forall u \in M.$$

Inspired by [3, 5], we propose the scheme

$$u^{k+1} = \phi_{c^k}(W(u^k, u^{k+1})), \quad c^k = c(u^k, u^{k+1}) \tag{8.2.6}$$

$$W(u^k, u^{k+1}) = \phi_{c^k}^{-1}(u^k) + h\,\overline{\Omega}(u^k, u^{k+1})\,\overline{\mathrm{grad}}H(u^k, u^{k+1}), \tag{8.2.7}$$

[1] To avoid cluttered notation we will just write $\overline{\mathrm{grad}}$ for the triple $(\overline{\mathrm{grad}}, \phi, c)$ in the sequel.

where h is the step size. The scheme (8.2.6)–(8.2.7) preserves the invariant H, since

$$\begin{aligned} H(u^{k+1}) - H(u^k) &= g(\overline{\operatorname{grad}} H(u^k, u^{k+1}), \phi_{c^k}^{-1}(u^{k+1}) - \phi_{c^k}^{-1}(u^k)) \\ &= g(\overline{\operatorname{grad}} H(u^k, u^{k+1}), h\,\overline{\Omega}(u^k, u^{k+1})\,\overline{\operatorname{grad}} H(u^k, u^{k+1})) = 0. \end{aligned}$$

Here and in the following we adopt the shorthand notation $c = c(u, v)$ as long as it is obvious what the arguments of c are.

The Average Vector Field (AVF) method has been studied extensively in the literature; some early references are [11, 19, 23]. This is a discrete gradient method, and we propose a corresponding DRG satisfying (8.2.4)-(8.2.5) as follows:

$$\overline{\operatorname{grad}}_{\mathrm{AVF}} H(u, v) = \int_0^1 (T_{\gamma_\xi}\phi_c)^{\mathrm{T}} \operatorname{grad} H(\phi_c(\gamma_\xi))\,\mathrm{d}\xi, \tag{8.2.8}$$

where $\gamma_\xi = (1-\xi)\phi_c^{-1}(u) + \xi\phi_c^{-1}(v)$ and $(T_x\phi_c)^{\mathrm{T}} : T_{\phi_c(x)}M \to T_xM$ is the unique operator satisfying

$$g((T_x\phi_c)^{\mathrm{T}} a, b) = g(a, T_x\phi_c\, b), \quad \forall x, b \in T_cM, \quad a \in T_{\phi_c(x)}M.$$

Furthermore, we have the generalization of Gonzalez' midpoint discrete gradient [9],

$$\overline{\operatorname{grad}}_{\mathrm{MP}} H(u,v) = \operatorname{grad} H(c(u,v)) + \frac{H(v) - H(u) - g(\operatorname{grad} H(c(u,v)), \eta)}{g(\eta,\eta)}\eta \tag{8.2.9}$$

where $\eta = \phi_c^{-1}(v) - \phi_c^{-1}(u)$.

Note that both these DRGs involve the gradient of the first integral. This may be a disadvantage if H is non-smooth or if its gradient is expensive to compute. Also, the implicit nature of the schemes requires the solution of an n-dimensional nonlinear system of equations at each time step. An alternative is to consider the Itoh–Abe discrete gradient [13], also called the coordinate increment discrete gradient [19], which in certain cases requires only the solution of n decoupled scalar equations. We now present a generalization of the Itoh–Abe discrete gradient to finite-dimensional Riemannian manifolds.

8.2.3 Itoh–Abe discrete Riemannian gradient

Definition 8.3. For any tangent space T_cM one can choose a basis $\{E_1, ..., E_n\}$ composed of tangent vectors E_i, $i = 1, ..., n$, orthonormal with respect to the Riemannian metric g. Then, given $u, v \in M$, there exists a unique $\{\alpha_i\}_{i=1}^n$ so that

$$\phi_c^{-1}(v) - \phi_c^{-1}(u) = \sum_{i=1}^n \alpha_i E_i.$$

The *Itoh–Abe DRG* of the first integral H is then given by

$$\overline{\text{grad}}_{\text{IA}} H(u,v) = \sum_{j=1}^{n} a_j E_j, \tag{8.2.10}$$

where

$$a_j = \begin{cases} \dfrac{H(w_j) - H(w_{j-1})}{\alpha_j} & \text{if } \alpha_j \neq 0, \\ g(\text{grad} H(w_{j-1}), T\phi_c(\eta_{j-1})E_j) & \text{if } \alpha_j = 0, \end{cases}$$

$$w_j = \phi_c(\eta_j), \quad \eta_j = \phi_c^{-1}(u) + \sum_{i=1}^{j} \alpha_i E_i.$$

We refer to [3] for proof that this is indeed a DRG satisfying (8.2.4)-(8.2.5).

8.2.4 Euclidean setting

Let $M = V$ be an $\mathbb{R}$-linear space, and let g be the Euclidean inner product. The operator Ω is a solution dependent skew-symmetric $n \times n$ matrix $\Omega(u)$. For any $u \in V$, we have $T_u V \equiv V$. The retraction $\phi : V \to V$ is defined as $\phi_p(x) = p + x$, the Riemannian exponential on V, so that $\phi_c^{-1}(v) - \phi_c^{-1}(u) = v - u$. The gradient $\text{grad} H$ is an n-vector whose ith component is $\frac{\partial H}{\partial u_i}$, and the definition of the discrete Riemannian gradient coincides with the standard discrete gradient, since (8.2.4) now reads

$$H(v) - H(u) = \overline{\text{grad}} H(u,v)^{\mathrm{T}}(v-u).$$

Furthermore, (8.2.6)-(8.2.7) simply becomes the discrete gradient method introduced in [9], given by the scheme

$$u^{k+1} - u^k = h\overline{\Omega}(u^k, u^{k+1})\,\overline{\text{grad}} H(u^k, u^{k+1}), \tag{8.2.11}$$

where $\overline{\Omega}$ is a skew-symmetric matrix approximating Ω. Typical choices are $\overline{\Omega}(u^k, u^{k+1}) = \Omega(u^k)$, or $\overline{\Omega}(u^k, u^{k+1}) = \Omega((u^{k+1} + u^k)/2)$ if one seeks a symmetric method.

The DRGs (8.2.8) and (8.2.9) become the standard AVF and midpoint discrete gradients in this case. For the Itoh–Abe DRG, the practical choice for the orthogonal basis would be the set of unit vectors, $\{e_1, ..., e_n\}$, so that $\alpha_i = v_i - u_i$, and we get (8.2.10) with

$$a_j = \begin{cases} \dfrac{H(w_j) - H(w_{j-1})}{v_j - u_j} & \text{if } u_j \neq v_j, \\ \frac{\partial H}{\partial u_j}(w_{j-1}) & \text{if } u_j = v_j, \end{cases}$$

$$w_j = \textstyle\sum_{i=1}^{j} v_i e_i + \sum_{i=j+1}^{n} u_i e_i,$$

which is a reformulation of the Itoh–Abe discrete gradient as it is given in [13], [19] and the literature otherwise.

8.3 Methods of higher order

In the Euclidean setting, a strategy to obtain energy preserving methods of higher order was presented in [2] and later in [10], see also [6]. This technique is generalized to a Lie group setting in [5]. We will here formulate these methods in the context of Riemannian manifolds.

8.3.1 Energy-preserving collocation-like methods on Riemannian manifolds

Let $\mathsf{c}_1, ..., \mathsf{c}_s$ be distinct real numbers. Consider the Lagrange basis polynomials,

$$l_i(\xi) = \prod_{j=1, j\neq i}^{s} \frac{\xi - \mathsf{c}_j}{\mathsf{c}_i - \mathsf{c}_j}, \quad \text{and let} \quad b_i := \int_0^1 l_i(\xi)\, d\xi. \tag{8.3.1}$$

We assume that $\mathsf{c}_1, \ldots, \mathsf{c}_s$ are such that $b_i \neq 0$ for all i. A step of the energy-preserving collocation-like method, starting at $u^0 \in M$, is defined via a polynomial $\sigma : \mathbb{R} \to T_c M$ of degree s satisfying

$$\sigma(0) = \phi_c^{-1}(u^0), \tag{8.3.2}$$

$$\frac{d}{d\xi}\sigma(\xi h)\Big|_{\xi=\mathsf{c}_j} = T_{U_j}\phi_c^{-1}\Big(\Omega_j \mathrm{grad}_j H\Big), \quad U_j := \phi_c\Big(\sigma(\mathsf{c}_j h)\Big) \tag{8.3.3}$$

$$u^1 := \phi_c\big(\sigma(h)\big), \tag{8.3.4}$$

where

$$\mathrm{grad}_j H := \int_0^1 \frac{l_j(\xi)}{b_j}\Big(T_{U_j}\phi_c^{-1}\Big)^{\mathrm{T}} (T_{\sigma(\xi h)}\phi_c)^{\mathrm{T}} \mathrm{grad} H\big(\phi_c(\sigma(\xi h))\big)\, \mathrm{d}\xi,\ \Omega_j := \Omega(U_j).$$

Notice that with $s = 1$ and independently on the choice of c_1, we reproduce the DRG method (8.2.6)-(8.2.7) with the AVF DRG (8.2.8).

Using Lagrange interpolation and (8.3.3), the derivative of $\sigma(\xi h)$ at every point ξh is

$$\frac{d}{d\xi}\sigma(\xi h) = \sum_{j=1}^{s} l_j(\xi) T_{U_j}\phi_c^{-1}\Big(\Omega_j \mathrm{grad}_j H\Big), \tag{8.3.5}$$

from which by integrating we get

$$\sigma(\tau h) = \phi_c^{-1}(u_0) + h\sum_{j=1}^{s}\int_0^{\tau} l_j(\xi)\,\mathrm{d}\xi\, T_{U_j}\phi_c^{-1}\Big(\Omega_j \mathrm{grad}_j H\Big).$$

The defined method is energy preserving, which we see by using

$$\frac{\mathrm{d}}{\mathrm{d}\xi}\left(\phi_c(\sigma(\xi h))\right) = T_{\sigma(\xi h)}\phi_c\left(\frac{\mathrm{d}}{\mathrm{d}\xi}\sigma(\xi h)\right),$$

and (8.3.5) to get

$$\begin{aligned}
H(u^1) - H(u^0) &= \int_0^1 g\left(\mathrm{grad}H\left(\phi_c(\sigma(\xi h))\right), \frac{\mathrm{d}}{\mathrm{d}\xi}\phi_c(\sigma(\xi h))\right)\mathrm{d}\xi \\
&= \int_0^1 g\left(\mathrm{grad}H\left(\phi_c(\sigma(\xi h))\right), T_{\sigma(\xi h)}\phi_c\left(\sum_{j=1}^{s} l_j(\xi)\, T_{U_j}\phi_c^{-1}\left(\Omega_j \mathrm{grad}_j H\right)\right)\right)\mathrm{d}\xi \\
&= \int_0^1 g\left(\left(T_{\sigma(\xi h)}\phi_c\right)^{\mathrm{T}}\mathrm{grad}H\left(\phi_c(\sigma(\xi h))\right), \sum_{j=1}^{s} l_j(\xi)\, T_{U_j}\phi_c^{-1}\left(\Omega_j \mathrm{grad}_j H\right)\right)\mathrm{d}\xi \\
&= \sum_{j=1}^{s} b_j g\left(\int_0^1 \frac{l_j(\xi)}{b_j}\left(T_{U_j}\phi_c^{-1}\right)^{\mathrm{T}}\left(T_{\sigma(\xi h)}\phi_c\right)^{\mathrm{T}}\mathrm{grad}H\left(\phi_c(\sigma(\xi h))\right)\mathrm{d}\xi, \Omega_j \mathrm{grad}_j H\right) \\
&= \sum_{j=1}^{s} b_j g\left(\mathrm{grad}_j H, \Omega_j\, \mathrm{grad}_j H\right) = 0,
\end{aligned}$$

and hence repeated use of (8.3.2)-(8.3.4) ensures $H(u^k) = H(u^0)$ for all $k \in \mathbb{N}$.

8.3.2 Higher order extensions of the Itoh–Abe DRG method

From the Itoh–Abe DRG one can get a new DRG, also satisfying (8.2.4), by

$$\overline{\mathrm{grad}}_{\mathrm{SIA}} H(u,v) = \frac{1}{2}\left(\overline{\mathrm{grad}}_{\mathrm{IA}} H(u,v) + \overline{\mathrm{grad}}_{\mathrm{IA}} H(v,u)\right). \tag{8.3.6}$$

We call this the *symmetrized Itoh–Abe DRG*. Note that we need the base point c to be the same in the evaluation of $\overline{\mathrm{grad}}_{\mathrm{IA}} H(u,v)$ and $\overline{\mathrm{grad}}_{\mathrm{IA}} H(v,u)$. When $c(u,v) = c(v,u)$ and $\overline{\Omega}_{(u,v)} = \overline{\Omega}_{(v,u)}$, we get a symmetric DRG method (8.2.6)-(8.2.7), which is therefore of second order.

Alternatively, one can get a symmetric 2-stage method by a composition of the Itoh–Abe DRG method and its adjoint. Furthermore, one can get energy preserving methods of any order using a composition strategy. To ensure symmetry of an s-stage composition method, one needs $c_i(u,v) = c_{s+1-i}(v,u)$ for different center points c_i belonging to each stage and, similarly, $\overline{\Omega}_i(u,v) = \overline{\Omega}_{s+1-i}(v,u)$.

8.4 Error analysis

8.4.1 Local error

In this section, $\varphi_t(u)$ is the t-flow of the ODE vector field F. The most standard discrete gradient methods have a low or moderate order of convergence, and that is also the case for the DRG methods unless special care is taken in designing $\overline{\Omega}$ and $\overline{\mathrm{grad}}H$. We shall not pursue this approach here, but refer to the collocation-like methods if high order of accuracy is required. We shall see, however, that the methods designed here are consistent and can be made symmetric. Analysis of the local error can be done in local coordinates, assuming that the step size is always chosen sufficiently small, so that within a fixed step, $u^k, u^{k+1}, c(u^k, u^{k+1})$ and the exact local solution $u(t_{k+1})$ all belong to the same given coordinate chart. From the definition (8.2.6)-(8.2.7) it follows immediately that the representation of $u^{k+1}(h)$ satisfies $u^{k+1}(0) = u^k$ and $\frac{d}{dh}u^{k+1}(0) = F(u^k)$. Then by equivalence of local coordinate norms and the Riemannian distance, we may conclude that the local error in DRG methods satisfies

$$d(u^{k+1}, \varphi_h(u^k)) \le Ch^2.$$

Similar to what was also observed in [5], the DRG methods (8.2.6)-(8.2.7) are symmetric whenever $\overline{\mathrm{grad}}H(u,v) = \overline{\mathrm{grad}}H(v,u)$, $\overline{\Omega}(u,v) = \overline{\Omega}(v,u)$, and $c(u,v) = c(v,u)$ for all $u, v \in M$. In that case we obtain an error bound for the local error of the form $d(u^{k+1}, \varphi_h(u^k)) \le Ch^3$.

The collocation-like methods of section 8.3 have associated nodes $\{\mathsf{c}_i\}_{i=1}^s$ and weights $\{b_i\}_{i=1}^s$ defined by (8.3.1). The order of the local error depends on the accuracy of the underlying quadrature formula given by these nodes and weights. The following result is a simple consequence of Theorem 4.3 in [6].

Theorem 8.1. *Let ψ_h be the method defined by* (8.3.2)-(8.3.4)*. The order of the local error is at least*

$$p = \min(r, 2r - 2s + 2)$$

where r is the largest integer such that $\sum_{i=1}^s b_i c_i^{q-1} = \frac{1}{q}$ for all $1 \le q \le r$. This means that there are positive constants C and h_0 such that

$$d(\psi_h(u), \varphi_h(u)) \le C\, h^{p+1} \quad \text{for } h < h_0,\ u \in M.$$

Proof. Choose h small enough such that the solution can be represented in the form $u(h\xi) = \phi_c(\gamma(\xi h)), \xi \in [0,1]$, and consider the corresponding differential equation for γ in T_cM:

$$\frac{d}{dt}\gamma(t) = \left(\phi_c^* F\right)(\gamma(t)) = \left(T_{\gamma(t)}\phi_c\right)^{-1} \Omega\, \mathrm{grad}H\left(\phi_c(\gamma(t))\right). \tag{8.4.1}$$

Notice that $\left(T_\gamma \phi_c\right)^{-1} = T_U \phi_c^{-1}$ where $U = \phi_c \circ \gamma$ and $T_{U(t)}\phi_c^{-1} : T_{U(t)}M \to T_c M$ for every t. We obtain

$$\frac{d}{dt}\gamma(t) = T_{U(t)}\phi_c^{-1}\Omega\left(T_{U(t)}\phi_c^{-1}\right)^{\mathrm{T}}\left(T_{\gamma(t)}\phi_c\right)^{\mathrm{T}}\mathrm{grad}H\left(\phi_c(\gamma(t))\right). \tag{8.4.2}$$

Considering the Hamiltonian $\widetilde{H} : T_c M \to \mathbb{R}$, $\widetilde{H}(\gamma) := \phi_c^* H(\gamma) = H \circ \phi_c(\gamma)$, we can then rewrite (8.4.1) in the form

$$\frac{d}{dt}\gamma(t) = \widetilde{\Omega}(\gamma)\,\mathrm{grad}\widetilde{H}(\gamma), \quad \widetilde{\Omega}(\gamma) := T_{U(t)}\phi_c^{-1}\Omega\left(T_{U(t)}\phi_c^{-1}\right)^{\mathrm{T}}, \tag{8.4.3}$$

where we have used that $\mathrm{grad}\widetilde{H} = T_{\gamma(t)}\phi_c^{\mathrm{T}}\mathrm{grad}H\left(\phi_c(\gamma(t))\right)$, which is now a gradient on the linear space $T_c M$ with respect to the metric inherited from M, $g|_c$. Locally in a neighborhood of c, (8.3.2)-(8.3.4) applied to (8.4.3) coincides with the methods of Cohen and Hairer, and therefore the order result [6, Thm 4.3] can be applied. Since the Riemannian distance $d(\cdot,\cdot)$ and any norm in local coordinates are equivalent, the result follows. □

8.4.2 Global error

We prove the following result for the global error in DRG methods.

Theorem 8.2. *Let $u(t)$ be the exact solution to (8.2.1) where F is a complete vector field on a connected Riemannian manifold (M, g) with flow $u(t) = \varphi_t(u^0)$. Let ψ_h represent a numerical method $u^{k+1} = \psi_h(u^k)$ whose local error can be bounded for some $p \in \mathbb{N}$ as*

$$d(\psi_h(u), \varphi_h(u)) \le Ch^{p+1} \quad \textit{for all } u \in M.$$

Suppose there is a constant L such that

$$\|\nabla F\|_g \le L,$$

where ∇ is the Levi-Civita connection and $\|\cdot\|_g$ is the operator norm with respect to the metric g. Then the global error is bounded as

$$d\left(u(kh), u^k\right) \le \frac{C}{L}(\mathrm{e}^{khL} - 1)h^p \quad \textit{for all } k > 0.$$

Proof. Denoting the global error as $e^k := d(u(kh), u^k)$, the triangle inequality yields

$$e^{k+1} \le d\left(\varphi_h(u(kh)), \varphi_h(u^k)\right) + d\left(\varphi_h(u^k), \psi_h(u^k)\right).$$

The first term is the error at nh propagated over one step, the second term is the local error. For the first term, we find via a Grönwall type inequality of [14],

$$d\left(\varphi_h(u(kh)), \varphi_h(u^k)\right) \le \mathrm{e}^{hL} d\left(u(kh), u^k\right) = \mathrm{e}^{hL} e^k.$$

Using the local error estimate for the second term, we get the recursion

$$e^{k+1} \leq \mathrm{e}^{hL} e^k + Ch^{p+1},$$

which yields

$$e^k \leq C \frac{\mathrm{e}^{khL} - 1}{\mathrm{e}^{hL} - 1} h^{p+1} \leq \frac{C}{L} (\mathrm{e}^{khL} - 1) h^p.$$

□

Remark: Following Theorem 1.4 in [14], the condition that F is complete can be relaxed if $\varphi_t(u^0)$ and $\{u^k\}_{k\in\mathbb{N}}$ lie in a relatively compact submanifold N of M containing all the geodesics from u^k to $\varphi_{kh}(u^0)$. This is the case if, for instance, H has compact, geodesically convex sublevel sets, since both $\varphi_t(u^0)$ and $\{u^k\}_{k\in\mathbb{N}}$ are restricted to the level set $M_{H(u^0)} = \{p \in M \,|\, H(p) = H(u^0)\}$ and hence lie in the sublevel set $N_{H(u^0)} = \{p \in M \,|\, H(p) \leq H(u^0)\}$.

8.5 Examples and numerical results

We test our methods on two different variants of the classical spin system, whose solution evolves on the d-fold product of two-spheres, $(S^2)^d$,

$$\frac{\mathrm{d}s_i}{\mathrm{d}t} = s_i \times \frac{\partial H}{\partial s_i}, \quad s_i \in S^2, \quad i = 1, \ldots, d, \quad H \in \mathcal{F}\left(\left(S^2\right)^d\right). \tag{8.5.1}$$

The Riemannian metric g on $(S^2)^d$ restricts to the so-called round metric on each copy of the sphere. This metric coincides with the Euclidean inner product on the tangent planes of each of the spheres.

Geometric integrators for such systems are discussed widely in the literature, see e.g. [8, 16–18] and references therein. We study one or more bodies whose orientation is represented by a vector s_i of unit length in $\mathbb{R}^3$, so that s_i lies on the manifold $M = S^2 = \left\{s \in \mathbb{R}^3 : \|s\| = 1\right\}$. Here and in what follows, $\|\cdot\|$ denotes the 2-norm. Starting with $d = 1$, our choice of retraction ϕ is given by its restriction to p,

$$\phi_p(x) = \frac{p + x}{\|p + x\|}, \tag{8.5.2}$$

with the inverse

$$\phi_p^{-1}(u) = \frac{u}{p^{\mathrm{T}} u} - p$$

defined when $p^{\mathrm{T}} u > 0$. We note that $p^{\mathrm{T}} x = 0$ for all $x \in T_p S^2$. The tangent map of the retraction and its inverse are given by

$$T_x \phi_p = \frac{1}{\|p + x\|} \left(I - \frac{(p + x) \otimes (p + x)}{\|p + x\|^2} \right), \; T_u \phi_p^{-1} = \frac{1}{p^{\mathrm{T}} u} \left(I - \frac{u \otimes p}{p^{\mathrm{T}} u} \right), \tag{8.5.3}$$

where $\otimes$ denotes the outer product[2] of the vectors. For $d > 1$, we use the retraction defined by $\Phi_p(x) = (\phi_{p_1}(x_i), \ldots, \phi_{p_d}(x_d))$, where each $\phi_{p_i}(x_i)$ is given by (8.5.2).

8.5.1 Example 1: Perturbed spinning top

We consider first a nonlinear perturbation of a spinning top, see [18]. This is a spin system with one spin s. Given the inertia tensor $\mathbb{I} = \mathrm{diag}(\mathbb{I}_1, \mathbb{I}_2, \mathbb{I}_3)$, and denoting by s^2 the component-wise square of s, we can define the Hamiltonian as

$$H(s) = \frac{1}{2}(\mathbb{I}^{-1}s)^{\mathrm{T}}(s + \frac{2}{3}s^2).$$

The ODE system can be written in the form

$$\frac{\mathrm{d}s}{\mathrm{d}t} = \Omega(s)\,\mathrm{grad}H(s), \quad \Omega(s) = \hat{s},$$

using the hat operator defined by $\hat{s}y = s \times y$. We approximate this system numerically, testing the scheme (8.2.6)-(8.2.7) with different discrete Riemannian gradients: the AVF (8.2.8), the midpoint (8.2.9), the Itoh–Abe (8.2.10) and its symmetrized version (8.3.6). For the three symmetric methods, we have chosen $c(s, \widetilde{s}) = \frac{s+\widetilde{s}}{\|s+\widetilde{s}\|}$, so that $\phi_c^{-1}(\widetilde{s}) = -\phi_c^{-1}(s)$. Using that $\mathrm{grad}H(s) = \mathbb{I}^{-1}(s + s^2)$ and considering the transpose of $T_{\gamma_\xi}\phi_c$ from (8.5.3), the AVF DRG becomes

$$\begin{aligned}\overline{\mathrm{grad}}_{\mathrm{AVF}}H(s,\widetilde{s}) &= \int_0^1 \frac{1}{\|l_\xi\|}\left(I - \frac{l_\xi \otimes l_\xi}{\|l_\xi\|^2}\right)\mathbb{I}^{-1}(\phi_c(\gamma_\xi) + \phi_c(\gamma_\xi)^2)\,\mathrm{d}\xi \\ &= \int_0^1 \frac{1}{\|l_\xi\|}\left(\mathbb{I}^{-1}\left(\phi_c(\gamma_\xi) + \phi_c(\gamma_\xi)^2\right) - \phi_c(\gamma_\xi)^{\mathrm{T}}\mathbb{I}^{-1}\left(\phi_c(\gamma_\xi) + \phi_c(\gamma_\xi)^2\right)\phi_c(\gamma_\xi)\right)\mathrm{d}\xi,\end{aligned}$$

with $\gamma_\xi = (1-\xi)\phi_c^{-1}(s) + \xi\phi_c^{-1}(\widetilde{s}) = (1-2\xi)\phi_c^{-1}(s)$ and $l_\xi = c + \gamma_\xi$. Similarly, the midpoint DRG becomes

$$\overline{\mathrm{grad}}_{\mathrm{MP}}H(s,\widetilde{s}) = \frac{\mathbb{I}^{-1}\left(s + \widetilde{s} + \frac{2}{3}\left(s^2 + s\widetilde{s} + \widetilde{s}^2\right)\right) + \frac{\frac{1}{2}\|s+\widetilde{s}\|^2 - 2}{\|\widetilde{s}-s\|^2}\left(H(\widetilde{s}) - H(s)\right)(\widetilde{s}-s)}{\|s+\widetilde{s}\|},$$

where we have used that $g(s,s) = s^{\mathrm{T}}s = 1$ for all $s \in S^2$. To obtain the basis of T_cM for the definition of the Itoh–Abe DRG, we have used the singular-value decomposition. For the first order scheme, noting that $\phi_s^{-1}(s) = 0$, we choose

[2] If x and y are in $\mathbb{R}^3$, $x \otimes y$ is the matrix-matrix product of x taken as a 3×1 matrix and y taken as a 1×3 matrix.

$c(s,\widetilde{s}) = s$, and get $\alpha_j = \phi_s(\widetilde{s})^{\mathrm{T}} E_j$, for $j = 1,2$. Then the DRG (8.2.10) can be written as

$$\overline{\mathrm{grad}}_{\mathrm{IA}} H(s,\widetilde{s}) = \frac{H(s') - H(s)}{\phi_s^{-1}(\widetilde{s})^{\mathrm{T}} E_1} E_1 + \frac{H(\widetilde{s}) - H(s')}{\phi_s^{-1}(\widetilde{s})^{\mathrm{T}} E_2} E_2, \tag{8.5.4}$$

where $s' = \phi_s((\phi_s^{-1}(\widetilde{s})^{\mathrm{T}} E_1) E_1)$.

We solve the same problem using the 4th, 6th and 8th order variants of the collocation-like scheme (8.3.2)-(8.3.4). Choosing in the 4th order case the Gaussian nodes $\mathsf{c}_{1,2} = \frac{1}{2} \mp \frac{\sqrt{3}}{6}$ as collocation points and setting $c(s,\widetilde{s}) = s$, we get the nonlinear system

$$\begin{aligned}
S_1 &= h\phi_{s_0}\left(\frac{1}{2} T_{S_1}\phi_{s_0}^{-1}\left(\Omega_1 \mathrm{grad}_1 H\right) + \left(\frac{1}{2} - \frac{\sqrt{3}}{3}\right) T_{S_2}\phi_{s_0}^{-1}\left(\Omega_2 \mathrm{grad}_2 H\right)\right),\\
S_2 &= h\phi_{s_0}\left(\left(\frac{1}{2} + \frac{\sqrt{3}}{3}\right) T_{S_1}\phi_{s_0}^{-1}\left(\Omega_1 \mathrm{grad}_1 H\right) + \frac{1}{2} T_{S_2}\phi_{s_0}^{-1}\left(\Omega_2 \mathrm{grad}_2 H\right)\right),\\
s_1 &= h\phi_{s_0}\left(T_{S_1}\phi_{s_0}^{-1}\left(\Omega_1 \mathrm{grad}_1 H\right) + T_{S_2}\phi_{s_0}^{-1}\left(\Omega_2 \mathrm{grad}_2 H\right)\right),
\end{aligned}$$

where

$$\begin{aligned}
\sigma(\xi h) = &\left(\left(3 + 2\sqrt{3}\right)\phi_{s_0}^{-1}(S_1) + \left(3 - 2\sqrt{3}\right)\phi_{s_0}^{-1}(S_2)\right)\xi\\
&+ \left(3\left(\sqrt{3} - 1\right)\phi_{s_0}^{-1}(S_2) - 3\left(1 + \sqrt{3}\right)\phi_{s_0}^{-1}(S_1)\right)\xi^2
\end{aligned}$$

and we use the transposes of (8.5.3) and $\mathrm{grad} H(s) = \mathbb{I}^{-1}(s + s^2)$ in the evaluation of $\mathrm{grad}_1 H$ and $\mathrm{grad}_2 H$. The 6th and 8th order schemes are derived in a similar manner, using the standard Gaussian nodes.

A second order scheme is derived by composing the Itoh–Abe DRG method with its adjoint, and a 4th order scheme is obtained by composing this method again with itself, as well as one by composition of the symmetrized Itoh–Abe DRG method with itself. In all stages of these composition methods, a symmetric $c(u,v)$ is used.

Plots confirming the order of all methods can be seen in Figure 8.1, where solutions using the different schemes are compared to a reference solution obtained using a very small step size. See the left hand panel of Figure 8.2 for numerical confirmation that our methods do indeed preserve the energy to machine precision, while the implicit midpoint method does not. In the right hand panel of Figure 8.2, the solution obtained by the Itoh–Abe DRG scheme with a step size $h = 1$ is plotted together with a solution obtained using the symmetrized Itoh–Abe DRG method with a much smaller time step. We

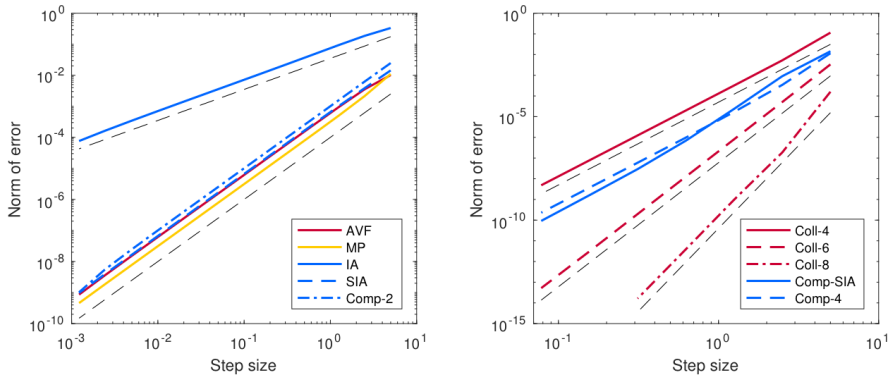


Figure 8.1: Error norm at $t = 10$ for the perturbed spinning top problem solved with different schemes, plotted with black, dashed reference lines of order 1, 2, 4, 6 and 8. Initial condition $s = (-1,-1,1)/\sqrt{3}$ and $\mathbb{I} = \mathrm{diag}(1,2,4)$. *Left:* The AVF, midpoint (MP), Itoh–Abe (IA) and symmetrized Itoh–Abe (SIA) DRGs and a 3-stage composition of the IA DRG scheme (Comp-2). *Right:* Collocation-type schemes of order 4, 6 and 8, a 3-stage composition of the SIA DRG scheme (Comp-SIA), and a 6-stage composition of the IA DRG scheme (Comp-4).

observe, as expected for a method that conserves both the energy and the angular momentum, that the solution stays on the trajectories of the exact solution, although not necessarily at the right place on the trajectory at any given time.

8.5.2 Example 2: Heisenberg spin chain

We now consider the Heisenberg spin chain of micromagnetics. This problem is considered in [8, 17], where different geometric integrators are tested. Here, $s \in \left(S^2\right)^d$, and the Hamiltonian is

$$H(s) = \sum_{i=1}^{d} {s_i}^{\mathrm{T}} s_{i-1}, \tag{8.5.5}$$

with $s_0 = s_d$ and $s_{d+1} = s_1$. The system (8.5.1) becomes, for this Hamiltonian,

$$\frac{\mathrm{d}s_i}{\mathrm{d}t} = \hat{s}_i \left(s_{i-1} + s_{i+1}\right), \quad i = 1,\dots,d,$$

and can be written in the block form

$$\frac{\mathrm{d}s}{\mathrm{d}t} = \Omega(s)\,\mathrm{grad} H(s), \quad \text{where} \quad \Omega(s) = \mathrm{diag}(\hat{s}_1,\dots,\hat{s}_d). \tag{8.5.6}$$

For such a d-particle system, we may write the DRGs as

$$\overline{\mathrm{grad}} H(s,\tilde{s}) = \left(\overline{\mathrm{grad}}^1 H(s,\tilde{s}),\dots,\overline{\mathrm{grad}}^d H(s,\tilde{s})\right),$$

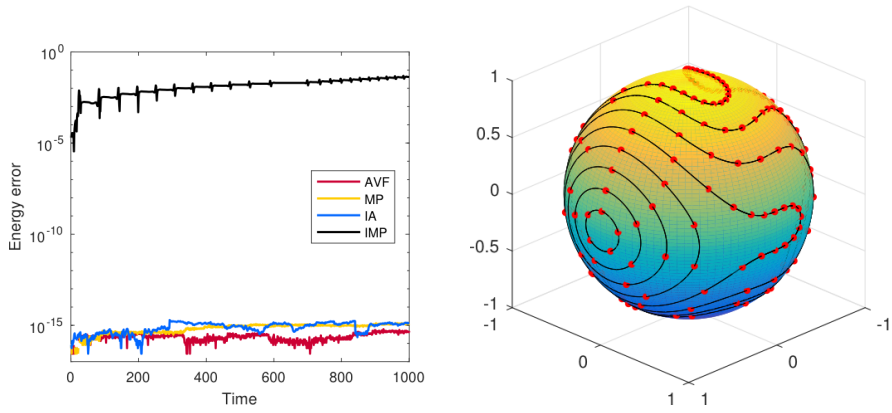


Figure 8.2: *Left:* Energy error with increasing time for the AVF, midpoint (MP) and Itoh–Abe (IA) DRG methods, as well as the implicit midpoint (IMP) method, with step size $h = 1$, initial condition $s = (-1,-1,1)/\sqrt{3}$ and $\mathbb{I} = \mathrm{diag}(1,2,4)$. *Right:* Curves of constant energy on the sphere, found by our method with different starting values. The black solid line is the solution using the symmetrized Itoh–Abe DRG method with step size $h = 0.01$, while the red dots are the solutions obtained by the Itoh–Abe DRG method with step size $h = 1$.

where we note that $\overline{\mathrm{grad}}^i H(s,\widetilde{s})$ is a discrete approximation to $\frac{\partial H}{\partial s_i}$. We thus get the AVF DRG defined by

$$\overline{\mathrm{grad}}^i_{\mathrm{AVF}} H(s,\widetilde{s}) = \int_0^1 \frac{1}{\|l_{i,\xi}\|}\left(I - \frac{l_{i,\xi}\otimes l_{i,\xi}}{\|l_{i,\xi}\|^2}\right)\Big(\phi_{c_{i-1}}(\gamma_{i-1,\xi}) + \phi_{c_{i+1}}(\gamma_{i+1,\xi})\Big)\,\mathrm{d}\xi$$

$$= \int_0^1 \frac{1}{\|l_{i,\xi}\|}\left(\phi_{c_{i-1}}(\gamma_{i-1,\xi}) + \phi_{c_{i+1}}(\gamma_{i+1,\xi}) - l_{i,\xi}^{\mathrm{T}}\Big(\phi_{c_{i-1}}(\gamma_{i-1,\xi}) + \phi_{c_{i+1}}(\gamma_{i+1,\xi})\Big)l_{i,\xi}\right)\mathrm{d}\xi,$$

with $\gamma_{i,\xi} = (1-2\xi)\phi_{c_i}^{-1}(s_i)$ and $l_{i,\xi} = c_i + \gamma_{i,\xi}$. For the midpoint DRG we get

$$\overline{\mathrm{grad}}^i_{\mathrm{MP}} H(s,\widetilde{s}) = c_{i-1} + c_{i+1} + \frac{H(\widetilde{s}) - H(s) - (\mathrm{grad}H(c(s,\widetilde{s}))^{\mathrm{T}}\eta}{\eta^{\mathrm{T}}\eta}\eta_i,$$

where $\eta = (\eta_1,\ldots,\eta_d)$ and $\eta_i = -2\phi_{c_i}^{-1}(s_i)$. In the numerical experiments, however, we have used a small modification of this,

$$\overline{\mathrm{grad}}^i_{\mathrm{MMP}} H(s,\widetilde{s}) = c_{i-1} + c_{i+1} + \frac{\widetilde{s}_i^{\mathrm{T}}\widetilde{s}_{i-1} - {s_i}^{\mathrm{T}}s_{i-1} - (c_{i-1}+c_{i+1})^{\mathrm{T}}\eta_i}{{\eta_i}^{\mathrm{T}}\eta_i}\eta_i.$$

This DRG, which does indeed satisfy (8.2.4)-(8.2.5), leads to a more computationally efficient scheme than the original midpoint DRG. Each $\overline{\mathrm{grad}}^i_{\mathrm{IA}} H(s,\widetilde{s})$ in the Itoh–Abe DRG is found as in the previous example, by (8.5.4). Higher order schemes are also derived in the same manner as before.

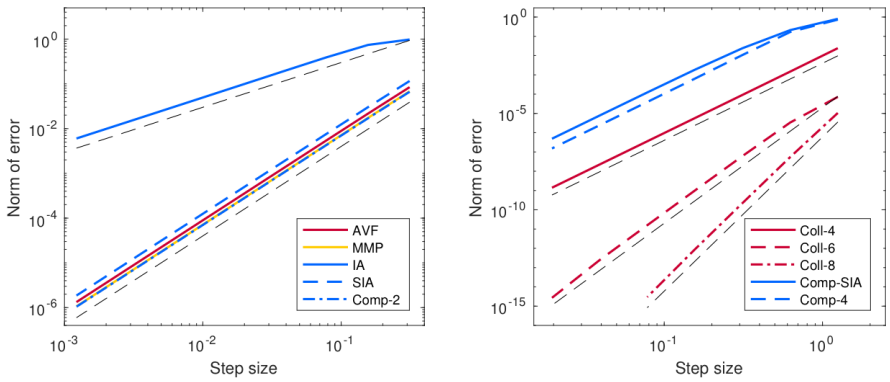


Figure 8.3: Error norm at $t = 10$ for the Heisenberg spin chain problem solved with different schemes, plotted with black, dashed reference lines of order 1, 2, 4, 6 and 8. *Left:* The AVF, modified midpoint (MMP), Itoh–Abe (IA) and symmetrized Itoh–Abe (SIA) DRGs and a 3-stage composition of the IA DRG scheme (Comp-2). *Right:* Collocation-type schemes of order 4, 6 and 8, a 3-stage composition of the SIA DRG scheme (Comp-SIA), and a 6-stage composition of the IA DRG scheme (Comp-4).

We test our schemes by comparing the numerical solutions with the exact solution

$$s_j(t) = (a\cos\theta_j + \tilde{a}\sin\theta_j)\cos\phi + \bar{a}\sin\phi, \quad \theta_j = jp - 2(1-\cos p)\sin\phi,$$

for a choice of constants $\phi, p \in \mathbb{R}$ and orthogonal unit vectors $a, \tilde{a}, \bar{a} \in \mathbb{R}^3$, see [8]. Order plots for the methods are provided in Figure 8.3, using $d = 5$, $\phi = \pi/3$, $p = 2\pi/d$, $a = (1,2,-1)/\sqrt{6}$, $\tilde{a} = (2,1,4)/\sqrt{21}$ and $\bar{a} = a \times \tilde{a}$. All schemes are shown to have the expected order.

8.6 Conclusions and further work

We have presented a general framework for constructing energy preserving numerical integrators on Riemannian manifolds. The main tool is to generalize the notion of discrete gradients as known from the literature. The new methods make use of an approximation to the Riemannian gradient coined the discrete Riemannian gradient, as well as a retraction map and a coordinate center function. An appealing feature of the new methods is that they do not depend on a particular choice of local coordinates or on an embedding of the manifold into a (larger) Euclidean space, but are of an intrinsic nature. Particular examples of discrete Riemannian gradient methods are given as generalizations of well-known schemes, such as the average vector field method, the midpoint discrete gradient method and the Itoh–Abe method. Extensions to higher order

are proposed via a collocation-like method. We have analysed the local and global error behaviour of the methods, and they have been implemented and tested for certain spin systems where the phase space is $\left(S^2\right)^d$.

Possible directions for future research include a more detailed study of the stability and propagation of errors, taking into account particular features of the Riemannian manifold; for instance, it may be expected that the sectional curvature will play an important role. More examples should also be tried out, and we belive, inspired by [3], that there is a potential for making our implementations more efficient by tailoring them for the particular manifold, as well as the ODE problem considered.

Bibliography

[1] R. L. ADLER, J.-P. DEDIEU, J. Y. MARGULIES, M. MARTENS, AND M. SHUB, *Newton's method on Riemannian manifolds and a geometric model for the human spine*, IMA J. Numer. Anal., 22 (2002), pp. 359–390.

[2] L. BRUGNANO, F. IAVERNARO, AND D. TRIGIANTE, *Hamiltonian boundary value methods (energy preserving discrete line integral methods)*, J. Numer. Anal. Ind. Appl. Math, 5 (2010), pp. 17–37.

[3] E. CELLEDONI, S. EIDNES, B. OWREN, AND T. RINGHOLM, *Dissipative schemes on Riemannian manifolds*, arXiv preprint arXiv:1804.08104, (2018).

[4] E. CELLEDONI, H. MARTHINSEN, AND B. OWREN, *An introduction to Lie group integrators – basics, new developments and applications*, J. Comput. Phys., 257 (2014), pp. 1040–1061.

[5] E. CELLEDONI AND B. OWREN, *Preserving first integrals with symmetric Lie group methods*, Discrete Contin. Dyn. Syst., 34 (2014), pp. 977–990.

[6] D. COHEN AND E. HAIRER, *Linear energy-preserving integrators for Poisson systems*, BIT, 51 (2011), pp. 91–101.

[7] P. E. CROUCH AND R. GROSSMAN, *Numerical integration of ordinary differential equations on manifolds*, J. Nonlinear Sci., 3 (1993), pp. 1–33.

[8] J. FRANK, W. HUANG, AND B. LEIMKUHLER, *Geometric integrators for classical spin systems*, J. Comput. Phys., 133 (1997), pp. 160–172.

[9] O. GONZALEZ, *Time integration and discrete Hamiltonian systems*, J. Nonlinear Sci., 6 (1996), pp. 449–467.

[10] E. HAIRER, *Energy-preserving variant of collocation methods*, JNAIAM. J. Numer. Anal. Ind. Appl. Math., 5 (2010), pp. 73–84.

[11] A. HARTEN, P. D. LAX, AND B. VAN LEER, *On upstream differencing and Godunov-type schemes for hyperbolic conservation laws*, SIAM Rev., 25 (1983), pp. 35–61.

[12] A. ISERLES, H. Z. MUNTHE-KAAS, S. P. NØRSETT, AND A. ZANNA, *Lie-group methods*, Acta Numer., 9 (2000), pp. 215–365.

[13] T. ITOH AND K. ABE, *Hamiltonian-conserving discrete canonical equations based on variational difference quotients*, J. Comput. Phys., 76 (1988), pp. 85–102.

[14] M. KUNZINGER, H. SCHICHL, R. STEINBAUER, AND J. A. VICKERS, *Global Gronwall estimates for integral curves on Riemannian manifolds*, Rev. Mat. Complut., 19 (2006), pp. 133–137.

[15] B. LEIMKUHLER AND G. W. PATRICK, *A symplectic integrator for Riemannian manifolds*, J. Nonlinear Sci., 6 (1996), pp. 367–384.

[16] D. LEWIS AND N. NIGAM, *Geometric integration on spheres and some interesting applications*, J. Comput. Appl. Math., 151 (2003), pp. 141–170.

[17] R. MCLACHLAN, K. MODIN, AND O. VERDIER, *A minimal-variable symplectic integrator on spheres*, Math. Comput., 86 (2017), pp. 2325–2344.

[18] R. I. MCLACHLAN, K. MODIN, AND O. VERDIER, *Symplectic integrators for spin systems*, Phys. Rev. E, 89 (2014).

[19] R. I. MCLACHLAN, G. R. W. QUISPEL, AND N. ROBIDOUX, *Geometric integration using discrete gradients*, Philos. T. R. Soc. A, 357 (1999), pp. 1021–1045.

[20] H. MUNTHE-KAAS, *Lie-Butcher theory for Runge-Kutta methods*, BIT, 35 (1995), pp. 572–587.

[21] H. MUNTHE-KAAS, *Runge-Kutta methods on Lie groups*, BIT, 38 (1998), pp. 92–111.

[22] H. Z. MUNTHE-KAAS AND A. LUNDERVOLD, *On post-Lie algebras, Lie–Butcher series and moving frames*, Found. Comput. Math., 13 (2013), pp. 583–613.

[23] G. QUISPEL AND D. MCLAREN, *A new class of energy-preserving numerical integration methods*, J. Phys. A: Math. Theor., 41 (2008).