Available online at www.sciencedirect.com
SCIENCE DIRECT®
Comput. Methods Appl. Mech. Engrg. 194 (2005) 511–530
Computer methods in applied mechanics and engineering
www.elsevier.com/locate/cma

Constrained interpolation (remap) of divergence-free fields

Pavel Bochev [a,*,1], Mikhail Shashkov [b,2]

[a] Computational Mathematics and Algorithms, Sandia National Laboratories, P.O. Box 5800, MS 1110, Albuquerque, NM 87185-1110, USA
[b] Theoretical Division, T-7, Los Alamos National Laboratory, Los Alamos, NM 87545, USA

Received 18 May 2004; accepted 18 May 2004

Abstract

A novel constrained interpolation algorithm for remapping of solenoidal face finite element vector fields is presented. The algorithm is based on explicit recovery, postprocessing and interpolation of a potential for the original vector field and a subsequent application of a curl operator to obtain the desired divergence-free finite element field on the new mesh.

The use of interpolation instead of advection in the remap process offers valuable computational advantages. Old and new meshes are neither required to have the same connectivity, nor to be close to each other. Slope limiting and upwinding, which can be sensitive to grid structure, are avoided and replaced by local optimization to control energy of the remapped field.

The new method is validated using a suite of cyclic remap problems on random and tensor product mesh sequences. A comparison with a local remapper based on a constrained transport advection algorithm is also included.

Keywords: Remap; Divergence-free; Exact sequence of finite element spaces

* Corresponding author. Tel.: +1 505 844 1990; fax: +1 505 845 7442.
E-mail addresses: pbboche@sandia.gov (P. Bochev), shashkov@lanl.gov (M. Shashkov).

[1] This work was funded by the Applied Mathematical Sciences program, US Department of Energy, Office of Energy Research and performed at Sandja National Labs, a multipro-gram laboratory operated by Sandia Corporation, a Lockheed-Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC-94AL85000.

[2] This work was performed under the auspices of the US Department of Energy at Los Alamos National Laboratory, under contract W-7405-ENG-36. This work was funded by the Applied Mathematical Sciences program, US Department of Energy, Office of Energy Research.

0045-7825/$ - see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.cma.2004.05.018

1. Introduction

Transfer of data between different grids, subject to constraints, is fundamental to many numerical algorithms. Prolongation and restriction operators in multilevel and adaptive methods, methods on non-matching grids, and computer simulations of problems with multiple physics and/or scales are just few examples that require this capability; see, for example, [2,8,10,15,19,22], and the references cited therein.

Another important example, that served to motivate a large part of this work, is arbitrary Lagrangian–Eulerian (ALE) methods [11]. These methods combine a Lagrangian update of the solution and the computational grid with rezoning and remapping phases wherein the grid distortion accrued during the Lagrangian motion is reduced, and the approximate solution is transferred to the improved mesh. A computational strategy that can combine the best properties of Eulerian and Lagrangian methods is to execute rezoning and remapping at every time cycle. The accuracy of the resulting continuous rezone ALE methods strongly depends on quality of the last, remapping phase and the availability of efficient and accurate remappers. A good remapper must also be robust and prevent pollution of solutions by unphysical features. For instance, remapping of concentrations or density fields must preserve positivity and total mass [19], while a magnetic flux **B** must remain divergence-free so as to avoid the spurious magnetic monopoles; see [3,23] for a discussion of the importance of this constraint in MHD.

In a continuous rezone ALE method individual grid movements can be limited to small perturbations of the initial mesh. To take advantage of this fact, remappers are often defined by adapting explicit advection algorithms [3] which use information only from the neighboring cells. However, connection between the advection equation and remapping of face element vector fields does not appear to be well-understood, in particular, the discretization errors engendered by advection remappers are not easily identified.

At the same time, it is clear that remapping represents an interpolation procedure that may be additionally constrained to provide physically meaningful solutions [19]. Association of remap with interpolation rather than with transport is more flexible because it allows formulation of algorithms on arbitrary pairs of grids, including grids with different topologies and grids that are not close to each other. If, on the other hand, the grids happen to be close and/or have the same connectivity, then constrained interpolation can be designed to take advantage from these features, making it potentially as efficient as an explicit transport based remapper. In addition to generality, constrained interpolation also allows to circumvent the often delicate and difficult issue of high order upwind interpolation and slope limiting for face element discretizations and unstructured grids [1,4], which are required in accurate transport algorithms.

In this paper we employ the constrained interpolation (CI) paradigm to develop a new remap algorithm for divergence-free finite element fields in two space dimensions without reference to advection. To avoid the costly solution of indefinite linear systems engendered by the use of Lagrange multipliers to enforce the constraint, our algorithm starts with an explicit recovery of a "vector" potential from the fluxes of a solenoidal field **B**. This process is a key component of the remapper and requires a discrete exactness property [6] to ensure the existence of discrete finite element potentials. The accuracy of the recovered potential is increased by an application of a postprocessing technique. A simple algorithm based on extension of the interpolation stencil along the cell faces is applied to provide more accurate values at face midpoints which are then used to compute an eight node serendipity representation of the potential. Then, local optimization is used to determine a convex combination of high and low order potentials that minimizes the energy mismatch between the original and the remapped fields. This optimized combination is interpolated to the target mesh, where application of the curl operator gives the desired divergence-free face element field. When the old and the new meshes have different connectivities and are not close to each other, interpolation requires global searches to locate the appropriate cells on the old mesh. However, in a continuous rezone

[3] This process can be reversed in the sense that a remap algorithm can be used to define an transport scheme; see [5].

ALE setting, only local searches will be necessary. In this case efficiency of a CI remapper is comparable to that of a transport based one.

The modular design of the CI remapper makes it very flexible and easy to specialize for different discretizations. The key, recovery, phase of the algorithm can be extended to any setting that provides an exact sequence of finite dimensional spaces, including mimetic finite differences [12–14], co-volume methods, or finite elements. The postprocessing phase can be implemented using interpolation techniques [16,17], polynomial preserving recovery [25,26], or reconstruction procedures from, e.g., finite volumes [18].

There are several important and novel aspects of our method that set it apart from the algorithms documented in the literature. Existing solutions that preserve exact divergence-free property are, as a rule, defined on Cartesian grids, and operate directly on the fluxes of the divergence-free field in a dimension by dimension basis; see e.g., [2,15,22]. One exception is [20] where a local remapper is defined by an application of a transport algorithm to a vector potential. Many of the existing algorithms also impose additional restrictions, such as grid hierarchy, or factor-of-two refinement. Interpolation on unstructured grids is considered in [8,10]. However, these papers adopt the technique of Lagrange multipliers to enforce the relevant constraint. This leads to a global saddle-point optimization problem for the remapped field that is not always easy to solve. In contrast, our approach eliminates the need for Lagrange multipliers by using potentials so that divergence-free condition is automatically satisfied by virtue of the definition of the remapped field as a curl. Reluctance to use potentials is perhaps due to the widely held opinion that their computation cannot be done in an effective explicit manner. However, as we demonstrate in this paper, such concerns are unfounded because potentials can be determined very efficiently.

To save time and space, in this paper we formulate and present the CI remapper for finite element spaces defined on logically rectangular grids. We validate the new method using a suite of two-dimensional cyclic remap example problems and compare it with a local remapper [21] derived from a finite element extension of the constrained transport (CT) algorithm of Evans and Hawley [9]. Numerical examples are selected to test critical aspects of the remap process such as handling of discontinuities and energy dissipation.

The paper is organized as follows. Section 2 introduces the relevant finite element spaces and reviews the exactness property that is fundamental to the explicit potential recovery. The remap problem is stated and solved in Section 3. Section 4 provides a brief description of a CT remapper. The paper concludes with a series of numerical experiments collected in Section 5 that demonstrate the performance of the new algorithm.

2. Finite element spaces

We define the finite element spaces used in this paper by a restriction of an exact sequence of finite elements defined with respect to an unstructured hexahedral mesh. For more details about the hexahedral elements and their construction we refer to [6,24].

2.1. Logically rectangular oriented meshes

Let $\Omega \in \mathbf{R}^2$ be an open bounded domain with polygonal boundary $\partial\Omega$, equipped with physical coordinates $(x_1, x_2) \equiv \mathbf{x}$. In what follows we restrict attention to domains that can be covered exactly by quadrilateral elements $\mathscr{K}$ arranged in a logically rectangular mesh $\mathscr{T}_h$ with nodes $\mathbf{x}_{i,j}$, $i = 1,\ldots,n$; $j = 1,\ldots,m$. The *horizontal* faces $\mathscr{F}_{i+1/2,j}$, of $\mathscr{T}_h$ connect adjacent nodes $(\mathbf{x}_{i,j}, \mathbf{x}_{i+1,j})$ that differ in their first index. The *vertical* faces $\mathscr{F}_{i,j+1/2}$ connect adjacent nodes $(\mathbf{x}_{i,j}, \mathbf{x}_{i,j+1})$ that differ in their second index. For $i = 1,\ldots,n-1$; $j = 1,\ldots,m-1$ each quadrilateral $\mathscr{K}_{i+1/2,j+1/2}$ in the mesh is associated with the four nodes $\mathbf{x}_{i,j}$, $\mathbf{x}_{i+1,j}$, $\mathbf{x}_{i,j+1}$, $\mathbf{x}_{i+1,j+1}$, and the four faces $\mathscr{F}_{i+1/2,j}$, $\mathscr{F}_{i+1/2,j+1}$, $\mathscr{F}_{i,j+1/2}$, $\mathscr{F}_{i+1,j+1/2}$; see Fig. 1. On a given element $\mathscr{K}_{i+1/2,j+1/2}$ we will also use the local indexing

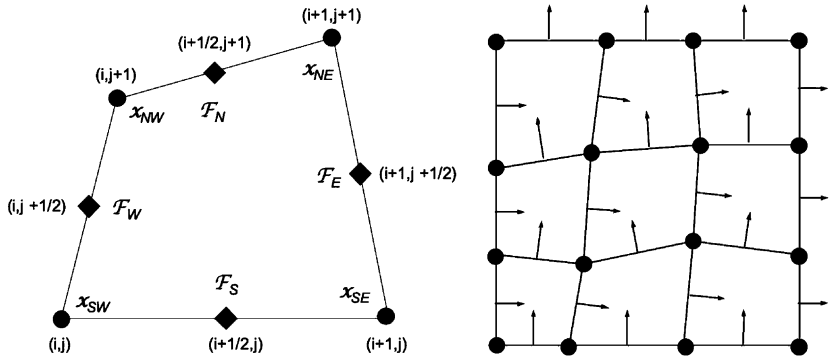

Fig. 1. Numbering and orientation choices for a logically rectangular grid.

$$\mathcal{F}_S = \mathcal{F}_{i+1/2,j}; \quad \mathcal{F}_N = \mathcal{F}_{i+1/2,j+1}; \quad \mathcal{F}_W = \mathcal{F}_{i,j+1/2}; \quad \mathcal{F}_E = \mathcal{F}_{i+1,j+1/2};$$

$$\mathbf{x}_{SW} = \mathbf{x}_{i,j}; \quad \mathbf{x}_{SE} = \mathbf{x}_{i+1,j}; \quad \mathbf{x}_{NW} = \mathbf{x}_{i,j+1}; \quad \mathbf{x}_{NE} = \mathbf{x}_{i+1,j+1}.$$

The sets of all nodes, faces and quadrilaterals in $\mathcal{T}_h$, are denoted by $\mathcal{N}(\mathcal{T}_h), \mathcal{F}(\mathcal{T}_h)$ and $\mathcal{K}(\mathcal{T}_h)$, respectively. Let $(\xi_1, \xi_2) \equiv \boldsymbol{\xi}$ denote a reference frame in $\mathbf{R}^2$. We assume that all quadrilaterals in $\mathcal{K}(\mathcal{T}_h)$ are strictly convex. Then; see [7], for every $\mathcal{K}_{i+1/2,j+1/2} \in \mathcal{K}(\mathcal{T}_h)$ there exists a unique bilinear map $F_{i+1/2,j+1/2} : \widehat{\mathcal{K}} \mapsto \mathcal{K}_{i+1/2,j+1/2}; \mathcal{F}_{i+1/2,j+1/2} = (F^1_{i+1/2,j+1/2}, F^2_{i+1/2,j+1/2})$ where $\widehat{\mathcal{K}} = [-1,1]^2$ is the *reference* element. The inverse map $G_{i+1/2,j+1/2} : \mathcal{K}_{i+1/2,j+1/2} \mapsto \widehat{\mathcal{K}}; G_{i+1/2,j+1/2} = (G^1_{i+1/2,j+1/2}, G^2_{i+1/2,j+1/2})$ is not polynomial unless $\mathcal{K}_{i+1/2,j+1/2}$ is a rectangle or a parallelepiped. For simplicity, if there's no chance for confusion, we will omit the indices and simply write F and G. For every $\boldsymbol{\xi} \in \widehat{K}$ and $\mathbf{x} \in \mathcal{K}$, the Jacobians $J_F(\boldsymbol{\xi})$ and $j_G(\mathbf{x})$ are invertible. The ith column of J_F will be denoted by $\mathbf{v}_i$, that is $J_F = (\mathbf{v}_1, \mathbf{v}_2)$.

The quadrilaterals and the faces in $\mathcal{T}_h$ are endowed with orientation as follows. Every $\mathcal{K}_{i+1/2,j+1/2} \in \mathcal{K}(\mathcal{T}_h)$ is oriented as a source, i.e., on its faces we choose the outward normal. The faces in $\mathcal{F}(\mathcal{T}_h)$ are oriented by selecting one of the two possible normal directions. This is done according to the kind of the face. For horizontal faces we choose the normal that runs along the down-up direction and for the vertical faces, the normal that runs along the left–right direction; see Fig. 1. Oriented faces are denoted by $(\mathcal{F}_U, \mathbf{n}_U)$ where $\mathbf{n}_U$ stands for the normal direction on face $\mathcal{F}_U$. The set of all oriented quadrilaterals forms a 2-chain and the boundary of each $\mathcal{K}_{i+1/2,j+1/2}$ is the 1-chain

$$\partial\mathcal{K} = -\mathcal{F}_S + \mathcal{F}_E + \mathcal{F}_N - \mathcal{F}_W = \sum_{U\in\{E,N,W,S\}} \sigma_U \mathcal{F}_U. \tag{1}$$

The symbol $-\mathcal{F}_U$ denotes the oriented face $(\mathcal{F}_U, \mathbf{n}_U)$. The multiplicities σ_U of the faces in the boundary chain take on the values ± 1 and reflect their orientations relative to the orientation of the quadrilateral $\mathcal{K}$. Orientation of quadrilaterals and faces in a mesh is not unique and can be defined in many different ways. The orientation choice used here is convenient for logically rectangular grids and aids in streamlining definitions of finite element functions and spaces.

2.2. Edge and face elements

To aid the discussion of finite element spaces and their properties it is convenient to view each quadrilateral $\mathcal{K}_{i+1/2,j+1/2}$ as planar projection of a hexahedral with unit height that was obtained by extruding $\mathcal{K}_{i+1/2,j+1/2}$ along the vector $\mathbf{k} = (0,0,1)$. The vertical edges of these hexahedrals are given by the segments

$(-0.5, 0.5)$ and their midpoints coincide with the nodes $\mathbf{x}_{i,j}$. The set of all such *virtual* edges is denoted by $\mathscr{E}(\mathscr{T}_h)$.

We first define the local element basis functions for a quadrilateral $\mathscr{K}$. There are four element basis functions associated with the four virtual edges:

$$\begin{aligned} W_{SW}(\mathbf{x}) &= \tfrac{1}{4}(1 - G^1(\mathbf{x}))(1 - G^2(\mathbf{x}))\mathbf{k}; \quad W_{SE}(\mathbf{x}) = \tfrac{1}{4}(1 + G^1(\mathbf{x}))(1 + G^2(\mathbf{x}))\mathbf{k}; \\ W_{NW}(\mathbf{x}) &= \tfrac{1}{4}(1 - G^1(\mathbf{x}))(1 + G^2(\mathbf{x}))\mathbf{k}; \quad W_{NE}(\mathbf{x}) = \tfrac{1}{4}(1 + G^1(\mathbf{x}))(1 + G^2(\mathbf{x}))\mathbf{k}. \end{aligned} \tag{2}$$

The element basis functions for the four faces of $\mathscr{K}$, are, see [6],

$$\begin{aligned} W_E(\mathbf{x}) &= \tfrac{1}{4}(1 + G^1(\mathbf{x}))(\nabla G^2(\mathbf{x}) \times \mathbf{K}); \quad W_W(\mathbf{x}) = \tfrac{1}{4}(1 - G^1(\mathbf{x}))(\nabla G^2(\mathbf{x}) \times \mathbf{K}); \\ W_N(\mathbf{x}) &= \tfrac{1}{4}(1 + G^2(\mathbf{x}))(\mathbf{k} \times \nabla G^1(\mathbf{x})); \quad W_S(\mathbf{x}) = \tfrac{1}{4}(1 - G^2(\mathbf{x}))(\mathbf{k} \times \nabla G^1(\mathbf{x})). \end{aligned} \tag{3}$$

The global basis functions for the virtual edges $\mathscr{E}(\mathscr{T}_h)$ and the faces $\mathscr{E}(\mathscr{T}_h)$ are defined in the usual manner by combining element basis functions from elements that share a virtual edge and a face, respectively. We will use the symbols W or W_j to denote the basis functions associated with an edge $\mathscr{E} \in \mathscr{E}(\mathscr{T}_h)$, and $W_{\mathscr{E}}$ for the basis functions associated with a face $\mathscr{F} \in \mathscr{F}(\mathscr{T}_h)$. These functions have the property that

$$\int_{-0.5}^{0.5} W_{i,j}(\mathbf{x}_{k,l})\mathrm{d}z = \delta_{kl}^{ij} \quad \text{and} \quad \int_{\mathscr{F}_U} W_T \cdot \mathbf{n}_U \mathrm{d}S = \delta_T^U.$$

The edge and face finite element spaces are defined as follows:

$$W^1(\mathscr{T}_h) = \mathrm{span}\{W_{\mathscr{E}}; \mathscr{E} \in \mathscr{E}(\mathscr{T}_h)\} \quad \text{and} \quad W^2(\mathscr{T}_h) = \mathrm{span}\{W_{\mathscr{F}}; \mathscr{F} \in \mathscr{E}(\mathscr{T}_h)\}.$$

The functions in these spaces are piecewise polynomials only when all elements in $\mathscr{T}_h$ are rectangles or parallelepipeds.

2.3. Discrete exactness property

Given a planar vector field $\mathbf{B}$, its interpolant $\mathbf{B}_h \in W^2(\mathscr{T}_h)$ is defined by

$$\mathbf{B}_h = \sum_{\mathscr{F} \in \mathscr{F}(\mathscr{T}_h)} \Phi_{\mathscr{F}} W_{\mathscr{F}}(\mathbf{x}); \Phi_{\mathscr{F}} = \int_{\mathscr{F}} \mathbf{B} \cdot \mathbf{n}_{\mathscr{F}} \mathrm{d}S. \tag{4}$$

Using the Divergence Theorem and (1) it is easy to see that

$$\int_{\mathscr{K}} \nabla \cdot \mathbf{B}_h \,\mathrm{d}\mathbf{x} = \int_{\partial\mathscr{K}} \mathbf{B}_h \cdot \mathbf{n} \,\mathrm{d}S = -\Phi_S + \Phi_E + \Phi_N - \Phi_W = \sum_{U \in \{S,E,N,W\}} \sigma_U \Phi_U,$$

where σ_U are the face multiplicities from the formula (1) for $\partial\mathscr{K}$. A function $\mathbf{B}_h \in W^2(\mathscr{T}_h)$, is said to be *discretely divergence-free* if

$$\sum_{U \in \{S,E,N,W\}} \sigma_U \Phi_U = 0 \quad \forall \mathscr{K} \in \mathscr{K}(\mathscr{T}_h).$$

This definition is applicable to a wide range of discrete vector fields, including those arising in mimetic finite differences for which point values are not defined everywhere.

Let $\mathscr{K} \in \mathscr{K}(\mathscr{T}_h)$ has local nodes $\mathbf{x}_{SW}$, $\mathbf{x}_{SE}$, $\mathbf{x}_{NW}$ and $\mathbf{x}_{NE}$. Taking the curl of the functions in (2) and comparing the resulting expressions with the element face basis functions in (3) reveals that

$$\begin{aligned} \nabla \times W_{SW} &= W_S - W_W; \quad \nabla \times W_{SE} = -W_S - W_E \\ \nabla \times W_{NW} &= W_N + W_W; \quad \nabla \times W_{NE} = -W_N + W_E \end{aligned} \tag{5}$$

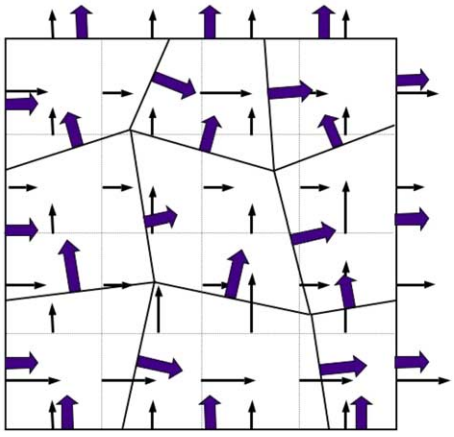

Fig. 2. The remap problem. Thin arrows represent the known fluxes on $\mathscr{T}_h^{\mathrm{o}}$. Thick arrows are the unknown fluxes on $\mathscr{T}_h^{\mathrm{n}}$.

that is, curls of element virtual edge basis functions are linear combinations of the element face basis functions. This relationship also extends to the sets of global basis functions so that we have the inclusion

$$\nabla \times W^1(\mathscr{T}_h) \subset W^2(\mathscr{T}_h). \tag{6}$$

3. Constrained interpolation (remap)

We consider a pair $\mathscr{T}_h^{\mathrm{o}}$ and $\mathscr{T}_h^{\mathrm{n}}$ of logically rectangular grids on Ω, that stand for the *old* and *new* partitions of Ω into quadrilateral elements. Except for the fact that both partitions are logically rectangular [4], no other relationship between them is assumed, e.g., they can have different numbers of elements. The problem of divergence-free remap from the old mesh $\mathscr{T}_h^{\mathrm{o}}$ into the new mesh $\mathscr{T}_h^{\mathrm{n}}$ is as follows (see Fig. 2).

Statement of the remap problem. Assume that $\mathbf{B}_h^{\mathrm{o}} \in W^2(\mathscr{T}_h^{\mathrm{o}})$ is a discretely divergence-free field on the old mesh. Find a discretely divergence-free approximation $\mathbf{B}_h^{\mathrm{n}} \in W^2(\mathscr{T}_h^{\mathrm{n}})$ of this field on the new mesh, such that

$$\int_\Omega |\mathbf{B}_h^{\mathrm{n}}|^2 \mathrm{d}\mathbf{x} = \int_\Omega |\mathbf{B}_h^{\mathrm{o}}|^2 \mathrm{d}\mathbf{x}. \tag{7}$$

The role of (7) is not to provide for a higher accuracy (very different fields can have identical energies), but to improve the quality of the remap step, assuming that $\mathbf{B}_h^{\mathrm{n}}$ is already a close approximation to $\mathbf{B}_h^{\mathrm{o}}$. In practice (7) may be very costly to enforce and in our algorithm we will opt for an inexact energy conservation.

3.1. Algorithm description

The main idea of the finite element remap algorithm is to recover, postprocess and interpolate a "vector" potential for $\mathbf{B}_h^{\mathrm{o}}$, and then take its curl on the new mesh to obtain a field $\mathbf{B}_h^{\mathrm{n}}$ on $\mathscr{T}_h^{\mathrm{n}}$. This will guarantee that $\nabla \cdot \mathbf{B}_h^{\mathrm{n}} = 0$, without using Lagrange multipliers.

[4] We recall that this assumption is made solely to simplify and shorten presentation of the method. The new and the old partitions can consist of different types of cells, e.g., triangles vs. quadrilaterals.

A key part in our strategy is played by the exactness (6) of the discrete spaces $W^1(\mathscr{T}_h)$ and $W^2(\mathscr{T}_h)$, respectively. This property ensures that every solenoidal field $\mathbf{B}_h^{\mathrm{o}} \in W^2(\mathscr{T}_h^{\mathrm{o}})$ has a discrete potential $\mathbf{A}_h^{\mathrm{o}} = (0,0,\phi_h^{\mathrm{o}}) \in W^1(\mathscr{T}_h^{\mathrm{o}})$, such that $\mathbf{B}_h^{\mathrm{o}} = \nabla \times \mathbf{A}_h^{\mathrm{o}}$. Moreover, we will show that this potential can be recovered without solving a linear system of equations. This makes our method as efficient as an advection based remapper because the cost of recovery does not exceed the cost of an explicit advection step.

Two other important components of the algorithm are the postprocessing and the interpolation operators, denoted by $\mathscr{P}$ and $\mathscr{I}$, respectively. However, definition of these operators is very flexible and they can be borrowed from other settings. Assuming that $\mathscr{P}$ and $\mathscr{I}$ are defined for the potential spaces $W^1(\mathscr{T}_h^{\mathrm{o}})$ and $W^1(\mathscr{T}_h^{\mathrm{n}})$, respectively, the constrained interpolation algorithm consists of the following main steps:

(1) **Recovery**. Given a solenoidal field $\mathbf{B}_h^{\mathrm{o}} \in W^2(\mathscr{T}_h^{\mathrm{o}})$ find $\mathbf{A}_h^{\mathrm{o}} \in W^1(\mathscr{T}_h^{\mathrm{o}})$, such that

$$\mathbf{B}_h^{\mathrm{o}} = \nabla \times \mathbf{A}_h^{\mathrm{o}}.$$

(2) **Postprocessing**. Define

$$\mathbf{A}_h(\lambda(\mathbf{x})) = \lambda(\mathbf{x})\mathbf{A}_h^{\mathrm{o}} + (1-\lambda(\mathbf{x}))(\mathscr{P}\mathbf{A}_{h,}^{\mathrm{o}}),$$

where $\lambda(\mathbf{x}): \Omega \mapsto [0,1]$ is a real valued function.

(3) **Optimization**. Solve the optimization problem

$$\lambda_{\mathrm{opt}}(\mathbf{x}) = \operatorname{argmin} \mathscr{J}(\nabla \times \mathbf{A}_h^{\mathrm{o}}, \nabla \times \mathscr{I}\mathbf{A}_h(\lambda(\mathbf{x}))),$$

where $\mathscr{J}$ is some measure of the energy mismatch of its arguments.

(4) **Remap**. Set

$$\mathbf{B}_h^{\mathrm{n}} = \nabla \times (\mathscr{I}\mathbf{A}_h(\lambda_{\mathrm{opt}}(\mathbf{x}))) \in W^2(\mathscr{T}_h^{\mathrm{n}}).$$

This remap algorithm may be readily extended to any discrete setting that has a discrete exactness property. If W^2 is a finite dimensional space used to represent the vector field $\mathbf{B}_h$, discrete exactness implies existence of spaces W^1, W^3 and operators $\mathbf{C}: W^1 \mapsto W^2$; $\mathbf{D}: W^2 \mapsto W^3$, such that

$$W^3 = \mathbf{D}(W^2) \quad \text{and} \quad \mathbf{B}_h = \mathbf{C}\mathbf{A}_h$$

for some $\mathbf{A}_h \in W^1$, whenever $\mathbf{D}\mathbf{B}_h = 0$. Consequently, the key assumption of our algorithm, i.e., existence of discrete vector potentials for discrete solenoidal fields, is satisfied. Definition of the remaining two components of the remapper can be adjusted to the choice of discrete spaces. For instance, the finite element algorithm that we are about to discuss, can be trivially extended to mimetic finite difference spaces, where the operators $\mathbf{C}$, and $\mathbf{D}$ are provided by the *natural* discretizations of the curl and the divergence; see [12–14].

3.2. Explicit recovery of the potential

In this section we present an algorithm that finds a potential for an arbitrary discretely divergence-free field $\mathbf{B}_h \in W^2(\mathscr{T}_h)$ without solving a linear system of equations. We seek $\mathbf{A}_h = (0,0,\phi_h) \in W^1(\mathscr{T}_h)$ such that $\nabla \times \mathbf{A}_h = \mathbf{B}_h$.

Let $\mathscr{K} \in \mathscr{K}(\mathscr{T}_h)$ be an arbitrary element. On this element the fluxes of $\mathbf{B}_h$ and the coefficients of $\mathbf{A}_h$ are related by $\nabla \times \mathbf{A}_h|_{\mathscr{K}} = \mathbf{B}_h|_{\mathscr{K}}$. Using (5)

$$(\nabla \times \mathbf{A}_h)|_{\mathscr{K}} = (\phi_{SW} - \phi_{SE})W_S + (\phi_{NW} - \phi_{SE})W_E + (\phi_{NW} - \phi_{NE})W_N + (\phi_{NW} - \phi_{SW})W_W.$$

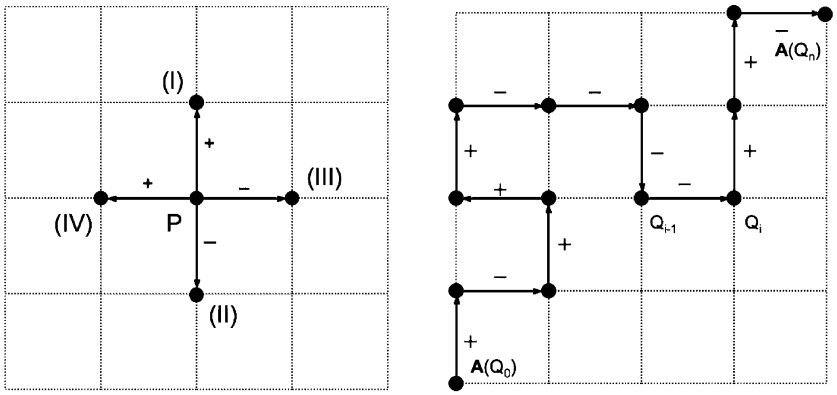

Fig. 3. Explicit recovery of a potential.

Comparing $\nabla \times \mathbf{A}_h$ and $\mathbf{B}_h$ on $\mathscr{K}$ gives four equations for the potential coefficients:

$$\begin{aligned} &\Phi_S = \phi_{SW} - \phi_{SE}; \quad \Phi_E = \phi_{NE} - \phi_{SE}, \\ &\Phi_N = \phi_{NW} - \phi_{NE}; \quad \Phi_W = \phi_{NW} - \phi_{SW}. \end{aligned} \tag{8}$$

These equations can be used to determine recursively the value of ϕ_h at any mesh point $Q \equiv \mathbf{x}_{i,j}$, provided an initial value (a gauge) is specified at some other arbitrary point $P \equiv \mathbf{x}_{i_0,j0}$. Consider first the case when P and Q are endpoints of a face $\mathscr{F}_U \in \mathscr{F}(\mathscr{T}_h)$. On logically rectangular grids P and Q can be in one of the following four configurations (see Fig. 3)

$$(P,Q) = \begin{cases} (\mathbf{x}_{i,j}, \mathbf{x}_{i,j+1}) & \text{case (I)}, \\ (\mathbf{x}_{i,j}, \mathbf{x}_{i,j-1}) & \text{case (II)}, \\ (\mathbf{x}_{i,j}, \mathbf{x}_{i+1,j}) & \text{case (III)}, \\ (\mathbf{x}_{i,j}, \mathbf{x}_{i-1,j}) & \text{case (IV)}. \end{cases}$$

We will define the face index μ_U of $\mathscr{F}_U$ depending on the configuration of P and Q

$$\mu_U = \begin{cases} 1 & \text{for configurations (I) and (IV)}, \\ -1 & \text{for configurations (II) and (III)}. \end{cases} \tag{9}$$

Using the face index, solution of (8) for $\phi_h(Q)$ can be expressed as

$$\phi_h(Q) = \phi_h(P) + \mu_U \Phi_U. \tag{10}$$

Consider now the general case where P and Q are arbitrary mesh points and $\phi_h(P)$ is given. Let $\{\mathscr{F}_{U_i}\}$; $i = 1, \ldots, n$ denote a subset of $\mathscr{F}(\mathscr{T}_h)$ that forms a path from P to Q and let $\{Q_i\}_{i=0}^{n}$ be the vertices along this path; see Fig. 3. For each face $\mathscr{F}_{U_i}$ we choose its index μ_{U_i} according to (9), and define the chain

$$C = \sum_{i=1}^{n} \mu_{U_i} \mathscr{F}_{U_i}.$$

A recursive application of (10) gives that

$$\phi_h(Q) = \phi_h(P) + \sum_{i=1}^{n} \mu_{U_i} \Phi_{U_i}. \tag{11}$$

Formula (11) defines the values of ϕ_h at all intermediate points Q_i on C, and so it can be used to recover the potential by following a path that passes through all points in the mesh. Next lemma shows that these values do not depend on the choice of this path.

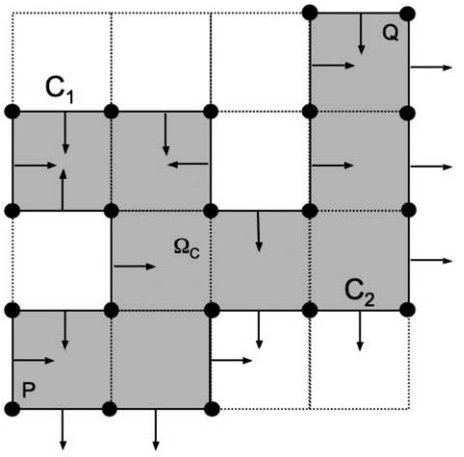

Fig. 4. Path independence of the recovered potential.

Lemma 1. *Assume that* $\mathbf{B}_h \in W^2(\mathscr{T}_h)$ *is discretely divergence-free. Let* P *denote an arbitrary but fixed point in* $\mathscr{T}_h$ *and let* Q *be some other grid point. Then, the value of* $\phi_h(Q)$*, computed according to* (11) *depends only on the value* $\phi_h(P)$*, but not on the path that connects the two points.*

Proof. Consider two different paths from P to Q and let

$$C_1 = \sum_{i=1}^{n_1} \mu_{U_i} \mathscr{F}_{U_i} \quad \text{and} \quad C_2 = \sum_{i=1}^{n_2} \mu_{U_i} \mathscr{F}_{U_i},$$

be the associated chains, where μ_{U_i} are determined according to (9). Let $\phi_h(Q_{C_1})$ and $\phi_h(Q_{C_2})$ denote the potential values at Q computed by application of (11) along the paths C_1 and C_2, respectively. Then,

$$\phi_h(Q_{C_1}) = \int_{C_1} \mathbf{B}_h \cdot \mathbf{n} \mathrm{d}S \quad \text{and} \quad \phi_h(Q_{C_2}) = \int_{C_2} \mathbf{B}_h \cdot \mathbf{n} \mathrm{d}S.$$

Without loss of generality we may assume that C_1 and C_2 are as shown in Fig. 4. Let Ω_C be the region enclosed by C. It is not hard to see that the effect of the face indices μ_{U_i} is to change the orientation of the normal on selected faces so that all face normals along C_1 point *inwards* the enclosed region, while all face normals along C_2 point *outwards* Ω_C. As a result, the boundary $\partial\Omega_C$ is given by the chain

$$C = C_2 - C_1.$$

Because $\mathbf{B}_h$ is discretely divergence free

$$0 = \int_{\Omega_C} \nabla \cdot \mathbf{B}_h \mathrm{d}\mathbf{x} = \int_C \mathbf{B}_h \cdot \mathbf{n} \mathrm{d}S = \int_{C_2 - C_1} \mathbf{B}_h \cdot \mathbf{n} \mathrm{d}S = \int_{C_2} \mathbf{B}_h \cdot \mathbf{n} \mathrm{d}S - \int_{C_1} \mathbf{B}_h \cdot \mathbf{n} \mathrm{d}S$$

and so we conclude that $\phi_h(Q_{C_1}) = \phi_h(Q_{C_2})$. □

To define the explicit potential recovery algorithm consider a path C_{sp} that forms a spanning tree for the graph built from the mesh faces. Let P denote the first node on this path. It is clear that by traversing C_{sp} once, the value of $\mathbf{A}_h$ will be determined at all mesh nodes $\mathbf{x}_{i,j}$ up to an arbitrary constant representing the value of $\phi_h(P)$. It is also clear that, without loss of generality, we can set $\phi_h(P) = 0$. Therefore, given a discrete solenoidal field $\mathbf{B}_h \in W^2(\mathscr{T}_h)$ the following algorithm finds its potential $\mathbf{A}_h \in W^1(\mathscr{T}_h)$:

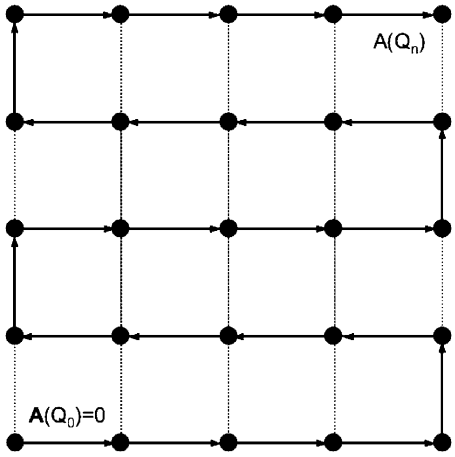

Fig. 5. Spanning tree for a logically rectangular mesh.

(1) Find a spanning tree for the faces $\mathscr{F}(\mathscr{T}_h)$ of $\mathscr{T}_h$ and form the chain

$$C_{sp} = \sum_{i=1}^{\mathrm{n}} \mu_{U_i} \mathscr{F}_{U_i},$$

where μ_{U_i} are defined according to (9).

(2) Set potential values according to

$$\phi_h(Q_0) = 0 \quad \text{and} \quad \psi_h(Q_s) = \phi_h(Q_{s-1}) + \mu_{U_s} \Phi_{U_s}; \quad s = 1, 2, \ldots, k,$$

where Q_s are the terminating points of the subchains

$$C_{sp}^{s} = \sum_{l=1}^{s} \mu_{U_l} \mathscr{F}_{U_l},$$

and Φ_{U_s} are the fluxes of $\mathbf{B}_h$ on the faces $\mathscr{F}_{S_s} = C_{sp}^{s}/C_{sp}^{s-1}$.

Upon completion, this algorithm generates a vector potential $\mathbf{A}_h \in W^1(\mathscr{T}_h)$ with the property that $\mathbf{V} \times \mathbf{A}_h = \mathbf{B}_h$. For logically rectangular grids a convenient spanning tree is shown in Fig. 5.

3.3. Postprocessing and interpolation

Consider the two partitions $\mathscr{T}_h^{\mathrm{o}}$ and $\mathscr{T}_h^{\mathrm{n}}$ of the computational domain and let $\mathbf{A}_h^{\mathrm{o}} = (0, 0, \phi_h^{\mathrm{o}})$ be the potential recovered from $\mathbf{B}_h^{\mathrm{o}}$. $\mathbf{A}_h^0$ can be readily used to find $\mathbf{A}_h^{\mathrm{n}} = (0, 0, \phi_h^{\mathrm{n}})$ on $\mathscr{T}_h^{\mathrm{n}}$, by computing the values of ϕ_h^{o} at the new nodes $\mathscr{N}(\mathscr{T}_h^{\mathrm{n}})$ and setting

$$\phi_{i,j}^{\mathrm{n}} = \phi_h^{\mathrm{o}}(\mathbf{x}_{i,j}^{\mathrm{n}}).$$

Evaluation of the right-hand side above requires us to find the element $\mathscr{K}_{k+1/2,l+1/2}^{\mathrm{o}}$ from the old mesh that contains $\mathbf{x}_{i,j}^{\mathrm{n}}$. If $\mathscr{T}_h^{\mathrm{n}}$ and $\mathscr{T}_h^{\mathrm{o}}$ are close to each other and have the same connectivity, this search will only require to check the elements that share the old node with the same index. For the grids considered in this paper there are at most 4 such elements.

However, differentiation of $\mathbf{A}_h^{\mathrm{n}}$ gives a solenoidal field $\mathbf{B}_h^{\mathrm{n}}$ that is only first-order accurate. To improve the accuracy of the remapper the finite element potential can be postprocessed. For examples of different finite

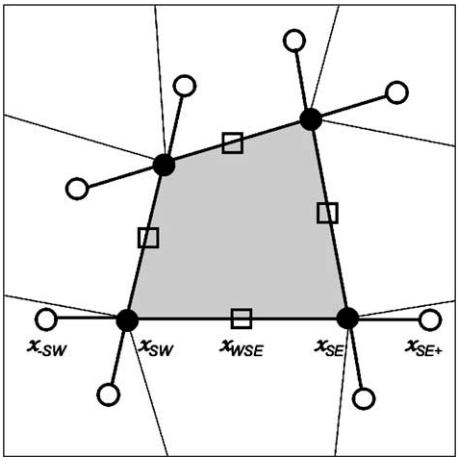

Fig. 6. Patch recovery procedure for the potential.

element postprocessing techniques we refer to [16,17], [25,26], among others. Here we devise a simple patch recovery scheme that is local and can be easily extended to other settings and element shapes.

For simplicity, let us consider a cell $\mathscr{K}^{\circ}_{i+1/2,j+1/2}$ from $\mathscr{K}(\mathscr{T}^{\circ}_h)$ that does not have a face on the boundary of the computational domain; see Fig. 6. The recovered potential $\mathbf{A}^{\circ}_h$ is defined by its four values at the cell vertices. In addition to these values we will compute four new values at the face midpoints. Consider for example the face $\mathscr{F}_S$ with endpoints $\mathbf{x}_{SW}$, $\mathbf{x}_{SE}$ and a midpoint $\mathbf{x}_{WSE}$. To define a value for $\mathbf{x}_{WSE}$ we proceed as follows. First, the face is extended in each direction by a fixed proportion of its length, say 1/4. Let $\mathbf{x}_{SW-}$ and $\mathbf{x}_{SE+}$ denote the endpoints of the extended face, so that with respect to the natural parametrization of the face by length

$$\mathbf{x}_{-SW} < \mathbf{x}_{SW} < \mathbf{x}_{WSE} < \mathbf{x}_{SE} < \mathbf{x}_{SE+}.$$

The values of ϕ°_h are given at $\mathbf{x}_{SW}$ and $\mathbf{x}_{SE}$, we proceed to compute the values of ϕ°_h at $\mathbf{x}_{-SW}$ and $\mathbf{x}_{SE+}$ by locating the cells that contain these endpoints and evaluating the local finite element expressions for $\mathbf{A}^{\circ}_h$. The set of four values along the extended face is interpolated by a cubic Lagrange interpolant. Finally, this polynomial is evaluated at the midpoint $\mathbf{x}_{WSE}$. This process is repeated for the remaining three faces to provide the desired midpoint values. If the cell $\mathscr{K}^{\circ}_{i+1/2,j+1/2}$ has one or two boundary faces, face extension is performed only in one direction. In this case, instead of a cubic, we use quadratic Lagrange interpolation to define the midpoint values.

Upon completion of the patch recovery process, on each element $\mathscr{K}$, there are eight values of the potential. These values are used to determine an 8-node serendipity interpolant of the potential on $\mathscr{K}$. Thus, the patch recovery operator $\mathscr{P}$ is defined by the process of Lagrange interpolation along element faces, followed by serendipity interpolation of vertex and midpoint values.

3.4. Optimization of the potential

To set up the optimization problem consider a function $\lambda(\mathbf{x}):\Omega \mapsto [0,1]$ and a convex combination

$$\mathbf{A}_h(\lambda(\mathbf{x})) = \lambda(\mathbf{x})\mathbf{A}^{\circ}_h + 1(1-\lambda(\mathbf{x}))\mathscr{P}\mathbf{A}^{\circ}_h, \tag{12}$$

of the reconstructed and postprocessed potentials. The idea is to determine $\lambda(\mathbf{x})$ so that the compound potential in (12) minimizes the *energy mismatch* functional

$$\mathscr{J}(\mathbf{B}_h^{\mathrm{o}}, \mathbf{B}_h(\lambda(\mathbf{x}))) = \sum_{\mathscr{K}\in\mathscr{K}(\mathscr{T}_h^{\mathrm{n}})} \left| \|\mathbf{B}_h^{\mathrm{o}}\|_{\mathscr{K}}^2 - \|\mathbf{B}_h(\lambda(\mathbf{x}))\|_{\mathscr{K}}^2 \right|^2,$$

where

$$\mathbf{B}_h(\lambda(\mathbf{x})) = \nabla \times (\mathscr{I}\mathbf{A}_h(\lambda(\mathbf{x})))$$

is the candidate solenoidal field on the new mesh. Let

$$\lambda_{\mathrm{opt}}(\mathbf{x}) = \operatorname{argmin} \mathscr{J}(\mathbf{B}_h^{\mathrm{o}}, \mathbf{B}_h^{\mathrm{n}}(\lambda(\mathbf{x}))) \tag{13}$$

be the optimal solution. The remapped field is defined by interpolating the optimized convex combination of potentials to the new mesh and then taking its curl:

$$\mathbf{B}_h^{\mathrm{n}} = \nabla \times (\mathscr{I}\mathbf{A}_h(\lambda_{\mathrm{opt}}(\mathbf{x}))). \tag{14}$$

In regions where $\mathbf{B}_h^{\mathrm{o}}$ does not have discontinuities or other sharp features, its energy will not experience rapid changes and the higher order component of $\mathbf{B}_h^{\mathrm{n}}(\lambda(\mathbf{x}))$ will tend to provide better approximation of the energy on the new mesh. As a result, in such regions $\lambda_{\mathrm{opt}}(\mathbf{x})$ will tend to 0 and the high order component $\mathscr{P}\mathbf{A}_h^{\mathrm{o}}$ and its curl will dominate in (12) and in (14), respectively.

In contrast, in regions where $\mathbf{B}_h^{\mathrm{o}}$ experiences sharp transitions and/or discontinuities, the presence of $\mathscr{P}\mathbf{A}_h^{\mathrm{o}}$ will tend to increase the energy of $\mathbf{B}_h^{\mathrm{n}}(\lambda(\mathbf{x}))$. As a result, in such regions minimization of the energy mismatch will tend to produce values of $\lambda(\mathbf{x})$ that are close to 1 and $\mathbf{B}_h^{\mathrm{n}}$ will be dominated by the curl of the low order component $\mathbf{A}_h^{\mathrm{o}}$. While $\lambda_{\mathrm{opt}}(\mathbf{x})$ is not a limiter in the sense that it does not guarantee monotonicity, its action resembles that of a limiter by increasing or decreasing the order of the approximation depending on the solution features.

To solve (13) efficiently we approximate $\lambda(\mathbf{x})$ by a piecewise constant function $\lambda(\mathscr{K})$. This choice of approximation effectively uncouples the global optimization problem into a set of local optimization problems

$$\lambda_{\mathrm{opt}}(\mathscr{K}) = \operatorname{argmin} \left| \|\mathbf{B}_h^{\mathrm{o}}\|_{\mathscr{K}}^2 - \|\mathbf{B}_h(\lambda(\mathscr{K}))\|_{\mathscr{K}}^2 \right|^2 \quad \forall \mathscr{K} \in \mathscr{K}(\mathscr{T}_h^{\mathrm{n}}) \tag{15}$$

for the element values $\lambda_{\mathrm{opt}}(\mathscr{K})$, that can be solved independently. For each problem we approximate the optimal value by using a fixed number of discrete bisection steps. After all element problems are solved, we obtain a discontinuous, piecewise constant approximation $\lambda_{\mathrm{opt}}(\mathscr{K})$ of $\lambda_{\mathrm{opt}}(\mathbf{x})$. Let $N(\mathbf{x}_{i,j})$ denote the number of elements that share $\mathbf{x}_{i,j}$ as a node. The piecewise constant function $\lambda(\mathscr{K})$ is further averaged to the nodes

$$\lambda(\mathbf{x}_{i,j}) = \left(\sum_{\mathscr{K}\ni\mathbf{x}_{i,j}} \lambda(\mathscr{K}) \right) \Big/ N(\mathbf{x}_{i,j}) \tag{16}$$

to define a nodal approximation of $\lambda_{\mathrm{opt}}(\mathbf{x})$. The final remapped field on the new mesh is obtained by using $\lambda(\mathbf{x}_{i,j})$ in (14). Implementation of this optimization strategy requires computation of the energy of the old field $\mathbf{B}_h^{\mathrm{o}}$ on the cells of the new mesh. One possibility is to compute $\|\mathbf{B}_h^{\mathrm{o}}\|_{0,\mathscr{K}}$ using quadrature. Another possibility is to treat energy as another quantity that needs to be remapped and use the sign-preserving conservative interpolation method from [19]. This will guarantee that the total energy of $\mathbf{B}_h^{\mathrm{o}}$ on the new mesh equals its total energy on the old mesh.

It is possible to simplify the optimization problem (15) even further by approximating $\lambda_{\mathrm{opt}}(\mathbf{x})$ by a global constant function $\lambda(\Omega)$. In this case, (13) reduces to an optimization problem

$$\lambda_{\mathrm{opt}}(\Omega) = \operatorname{argmin} \left| \|\mathbf{B}_h^{\mathrm{o}}\|_{\Omega}^2 - \|\mathbf{B}_h(\lambda(\Omega))\|_{\Omega}^2 \right|^2 \tag{17}$$

for the single value of $\lambda_{\text{opt}}(\Omega)$. In what follows we will refer to the strategy used in (15) as the *multiple parameter* optimization, while solution computed according to (17) will be referred to as the *single parameter* optimization.

Another possibility is to control energy mismatch by a feedback loop. In the simplest case we can use a single parameter λ^n determined according to

$$\lambda_n = \begin{cases} \max(0, \lambda^{\text{o}} + \varepsilon) & \varepsilon < 0, \\ \min(0, \lambda^{\text{o}} + \varepsilon) & \varepsilon > 0; \end{cases} \quad \varepsilon = \left(1 - \frac{\|\mathbf{B}_h^{\text{o}}\|_\Omega}{\|\mathbf{B}_h^{\text{n}}\|_\Omega}\right). \tag{18}$$

4. Advection based remap

In this section we briefly review the advection based remapper of [21] that will be used in the comparison tests. This remapper is based on finite element extension of the CT method [9] to logically rectangular grids. For extensions of CT type algorithms to simplicial triangulations using residual redistribution [1] ideas, see [4].

Throughout this section we assume that $\mathscr{T}_h^{\text{o}}$ and $\mathscr{T}_h^{\text{n}}$ have the same connectivity. Let $\mathbf{u}_{\text{rel}}$ denote the displacement field that takes the nodes of $\mathscr{T}_h^{\text{o}}$ into the nodes of $\mathscr{T}_h^{\text{n}}$. Given the old discretely divergence-free field $\mathbf{B}_h^{\text{o}} \in W^2(\mathscr{T}_h^{\text{o}})$, the new field $\mathbf{B}_h^{\text{n}} \in W^2(\mathscr{T}_h^{\text{n}})$ is approximated by marching the solution of the advection equation

$$\frac{\partial \mathbf{B}}{\partial t} = -\nabla \times (\mathbf{u}_{\text{rel}} \times \mathbf{B}) \quad \text{and} \quad \mathbf{B}(0, \mathbf{x}) = \mathbf{B}_h^{\text{o}} \tag{19}$$

forward in time by one unit time step. To apply this procedure, u_{rel} must be small enough to prevent a node in $\mathscr{T}_h^{\text{o}}$ from crossing more than one cell. To ensure that a CFL condition is satisfied we must restrict $\mathscr{T}_h^{\text{n}}$ to a small perturbation of $\mathscr{T}_h^{\text{o}}$. The finite difference equation for the new field is

$$\mathbf{B}_h^{\text{n}} = \mathbf{B}_h^{\text{o}} - \nabla \times (\mathbf{u}_{\text{rel}} \times \mathbf{B}_h^{\text{o}}). \tag{20}$$

The "electromotive force" $u_{\text{rel}} \times \mathbf{B}_h^{\text{o}}$ effected by the mesh displacement is approximated by

$$\mathbf{E}_h^{\text{o}} = \sum_{\mathscr{E} \in \mathscr{F}(\mathscr{T}_h^{\text{o}})} E_{\mathscr{E}}^{\text{o}} W_{\mathscr{E}}^{\text{o}} \in W^1(\mathscr{T}_h^{\text{o}}).$$

Using (5) it is easy to see that on each element $\mathscr{K}$

$$(\nabla \times \mathbf{E}_h^{\text{o}})|_{\mathscr{K}} = (E_{SW}^{\text{o}} - E_{SE}^{\text{o}}) W_S^{\text{o}} + (E_{NE}^{\text{o}} - E_{SE}^{\text{o}}) W_E^{\text{o}} + (E_{NW}^{\text{o}} - E_{NE}^{\text{o}}) W_N^{\text{o}} + (E_{NW}^{\text{o}} - E_{SW}^{\text{o}}) W_W^{\text{o}}.$$

As a result, on $\mathscr{K}$, the flux update (20) for the finite element field $\mathbf{B}_h^{\text{o}}$ reduces to

$$\begin{aligned} \Phi_S^{\text{n}} &= \Phi_S^{\text{o}}(E_{SW}^{\text{o}} - E_{SE}^{\text{o}}); \quad \Phi_E^{\text{n}} = \Phi_E^{\text{o}} + (E_{NE}^{\text{o}} - E_{SE}^{\text{o}}) \\ \Phi_N^{\text{n}} &= \Phi_N^{\text{o}}(E_{NW}^{\text{o}} - E_{NE}^{\text{o}}); \quad \Phi_W^{\text{n}} = \Phi_W^{\text{o}} + (E_{NW}^{\text{o}} - E_{SW}^{\text{o}}). \end{aligned} \tag{21}$$

The updated fluxes serve to define the vector field

$$\mathbf{B}_h^{\text{n}} = \sum_{\mathscr{F} \in \mathscr{F}(\mathscr{T}_h^{\text{n}})} \Phi_{\mathscr{F}}^{\text{n}} W_{\mathscr{F}}^{\text{n}} \in W^2(\mathscr{T}_h^{\text{n}})$$

relative to the new mesh $(\mathscr{T}_h^{\text{n}})$.

The flux update formulas (21) are the same as in the CT algorithm [9], except that they are defined for arbitrary unstructured quadrilateral cells. If $\nabla \cdot \mathbf{B}_h^{\text{o}} = 0$, (21) guarantees that $\mathbf{B}_h^{\text{n}}$ is also divergence-free, regardless of the manner in which the coefficients of $\mathbf{E}_h^{\text{o}}$ were computed. These coefficients are associated with the virtual edges where the finite element field $\mathbf{B}_h^{\text{o}}$ is discontinuous, and must be reconstructed in order

to evaluate $E^{\mathrm{o}}_{i,j}$. Because (19) is pure advection problem, reconstruction of $\mathbf{B}^{\mathrm{o}}_h$ at the nodes must use some form of upwinding. For logically rectangular grids a simple solution is to use the dimension by dimension upwind interpolant developed in [9]. For details of this procedure we refer to [9,21].

It is instructive to compare the CT remapper with the constrained interpolation algorithm. The recovered potential $\mathbf{A}^{\mathrm{o}}_h$ and its postprocessed version $\mathscr{P}\mathbf{A}^{\mathrm{o}}_h$ are analogues of low and high order reconstructions, respectively, in the CT remapper. The analogue of limiting in CI is provided by the optimization problem (13) that minimizes the energy mismatch between $\mathbf{B}^{\mathrm{o}}_h$ and its remapped version on the new mesh.

Even though the CT remapper was defined by an application of a transport scheme to an advection equation, in reality it is equivalent to a Taylor series approximation of the divergence-free field on the new mesh. To see this, consider a smooth solenoidal field $\mathbf{B}(\mathbf{x})$, a fixed point $\mathbf{x}_0$ and a small increment $\boldsymbol{\Delta}\mathbf{x} = (\Delta x, \Delta y)$. Then

$$\mathbf{B}^1(\mathbf{x}_0+\boldsymbol{\Delta}\mathbf{x}) = \mathbf{B}^1(\mathbf{x}_0) + \frac{\partial \mathbf{B}^1(\mathbf{x}_0)}{\partial x}\Delta x + \frac{\partial \mathbf{B}^1(\mathbf{x}_0)}{\partial y}\Delta y + \mathrm{O}(|\boldsymbol{\Delta}\mathbf{x}|^2)$$

$$\mathbf{B}^2(\mathbf{x}_0+\boldsymbol{\Delta}\mathbf{x}) = \mathbf{B}^2(\mathbf{x}_0) + \frac{\partial \mathbf{B}^2(\mathbf{x}_0)}{\partial x}\Delta x + \frac{\partial \mathbf{B}^2(\mathbf{x}_0)}{\partial y}\Delta y + \mathrm{O}(|\boldsymbol{\Delta}\mathbf{x}|^2)$$

Adding and subtracting $\frac{\partial \mathbf{B}^2(\mathbf{x}_0)}{\partial y}$ to the first equation, and $\frac{\partial \mathbf{B}^1(\mathbf{x}_0)}{\partial x}$ to the second equation and using that

$$\frac{\partial \mathbf{B}^1(\mathbf{x}_0)}{\partial x} + \frac{\partial \mathbf{B}^2(\mathbf{x}_0)}{\partial y} = 0$$

shows that the first terms in the Taylor series of $\mathbf{B}$ are

$$\mathbf{B}(\mathbf{x}_0+\boldsymbol{\Delta}\mathbf{x}) = \mathbf{B}(\mathbf{x}_0) - \nabla\times(\boldsymbol{\Delta}\mathbf{x}\times\mathbf{B}(\mathbf{x}_0)) + \mathrm{O}(|\boldsymbol{\Delta}\mathbf{x}|^2). \tag{22}$$

Similarity between this formula and the advection equation (20) is obvious. As a result, the CT remapper can be viewed as based on a *local interpolation* formula for the divergence-free field. The connection between (20) and the Taylor expansion (22) also indicates that the advection remapper will be second order accurate only when reconstruction of $\mathbf{B}^{\mathrm{o}}_h$ at the nodes leads to a first-order approximation of the derivatives.

5. Numerical examples

We test our algorithm using a cyclic remapping approach. This testing method was proposed in [19] and consists of remapping the function of interest on a sequence of grids parametrized by a "fictitious" time parameter $t \in [0,1]$. Therefore, we consider a sequence of grids $\{\mathscr{T}^{\mathrm{n}}_h\}_{n=0}^N$ where $\mathscr{T}^0_h = \mathscr{T}^N_h$, and the index n can be conveniently thought of as representing the fictitious time t^n. We begin with an initial solenoidal field $\mathbf{B}^0_h$, defined on $\mathscr{T}^0_h$, and then proceed to remap this field from $\mathscr{T}^n_h$ to $\mathscr{T}^{n+1}_h$ for $n = 0,\ldots,N-1$. Because the first and the last grids coincide, cyclic remap allows to inspect the cumulative effect of many remappings by comparing the initial and the final fields.

We use two different mesh sequences for the cyclic remap. The first sequence contains a set of 100 grids obtained by consecutive random perturbations, starting from a uniform initial grid $\mathscr{T}^{\mathrm{o}}_h$. The displacement field between $\mathscr{T}^{\mathrm{n}}_h$ and $\mathscr{T}^{n+1}_h$ is defined so that every grid in the sequence is guaranteed to be a valid quadrilateral partition of the unit square.

The second grid sequence is generated by using a displacement field given by

$$\begin{aligned} x_{i,j}(t) &= (1-\alpha(t))x_{i,j}(0) + \alpha(t)x^3_{i,j}(0),\\ y_{i,j}(t) &= (1-\alpha(t))y_{i,j}(0) + \alpha(t)y^2_{i,j}(0), \end{aligned} \tag{23}$$

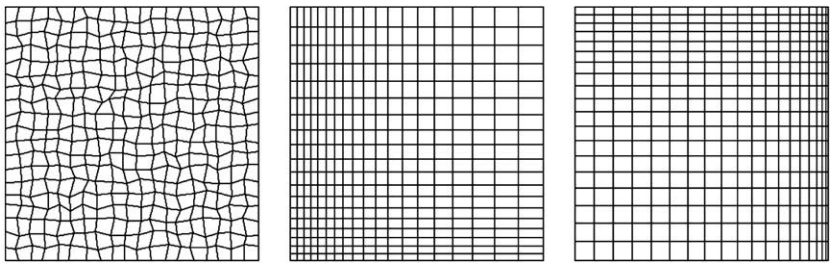

Fig. 7. Typical random mesh (left) and two snapshots of the tensor product mesh sequence.

where $\alpha = \sin(4\pi t)/2$ and $\mathbf{x}_{i,j}(0) = (x_{i,j}(0), y_{i,j}(0))$ are the node coordinates in the initial uniform grid $\mathscr{T}_h^{\mathrm{o}}$ The grids $\mathscr{T}_h^{\mathrm{n}}$ in this sequence are defined by $\mathbf{x}_{i,j}(t^n) = (x_{i,j}(t^n), y_{i,j}(t^n))$; $t^n = n/N$. One can prove that for any $0 \leqslant t \leqslant 1$ formulas (23) give a valid grid; see [19]. In contrast to the first sequence, these formulas produce a smooth displacement field, and a tensor product grid. Fig. 7 shows snapshots of the two mesh sequences.

The example solenoidal fields are defined by taking a curl of a potential function. This guarantees that they are divergence-free to machine precision. The first potential is $\mathbf{A} = (0, 0, \sin(2\pi x)\sin(2\pi y))$ so that

$$\mathbf{B} = (2\pi \sin(2\pi x)\cos(2\pi y), -2\pi \cos(2\pi x)\sin(2\pi y))^{\mathrm{T}}. \tag{24}$$

The second potential is a function that has a see-saw shape in x and is constant in y

$$\phi_h(\mathbf{x}) = \begin{cases} 4x & \text{if } 0 \leqslant x < 1/4, \\ -4(x - 0.5) & \text{if } 1/4 \leqslant x \leqslant 3/4, \\ 4(x-1) & \text{if } 3/4 < x \leqslant 0. \end{cases}$$

The curl of this potential gives a vector field

$$\mathbf{B}_1 = 0; \mathbf{B}_2 = \begin{cases} -4 & \text{if } 0 \leqslant x < 1/4, \\ 4 & \text{if } 1/4 \leqslant x \leqslant 3/4, \\ -4 & \text{if } 3/4 < x \leqslant 0 \end{cases} \tag{25}$$

with a discontinuous second component.

Our first experiment compares and contrasts different strategies in the computation of the parameter λ. We remap (24) and (25) using single and multiple parameter optimization, a feedback loop, and the fixed values $\lambda = 0$ and $\lambda = 1$. In the last two cases the remapper uses either the postprocessed, high-order potential ($\lambda = 0$), or the reconstructed, low-order one ($\lambda = 1$).

Fig. 8 shows energy levels of the remapped smooth field (24) on the two mesh sequences for different choices of λ. For $\lambda = 0$ and $\lambda = 1$ we see the typical growth and dissipation of energy associated with high and low order schemes, respectively. These figures also show little difference in the energy levels maintained by single parameter optimization (denoted by $\lambda(\Omega)$ in the plots) on one hand, and feedback loop control, on the other hand. The multiple parameter optimization (denoted by $\lambda(\mathscr{K})$ in the plots) tends to be a bit more dissipative for the random mesh sequence.

Fig. 9 shows profiles of the discontinuous field (25) on the last grid from the tensor product sequence, remapped by using multiple and single parameter optimization. While in both cases the remapper provides crisp profiles of the discontinuity, the use of a single parameter $\lambda(\Omega)$ leads to pronounced under and overshoots in the vicinity of the discontinuity. This is caused by the inability of a single parameter to account for the local behavior of the remapped field. In contrast, because the multiple parameter optimization can

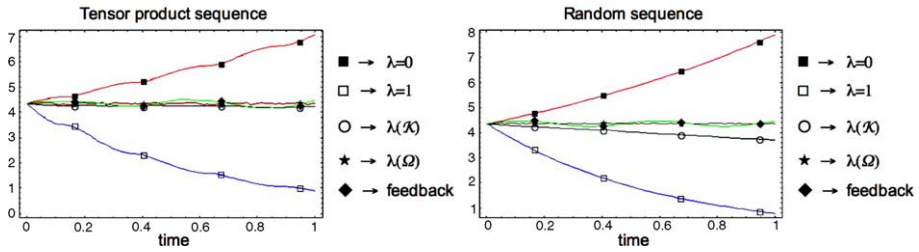

Fig. 8. Energy of the remapped field for different choices of λ and tensor product mesh sequence.

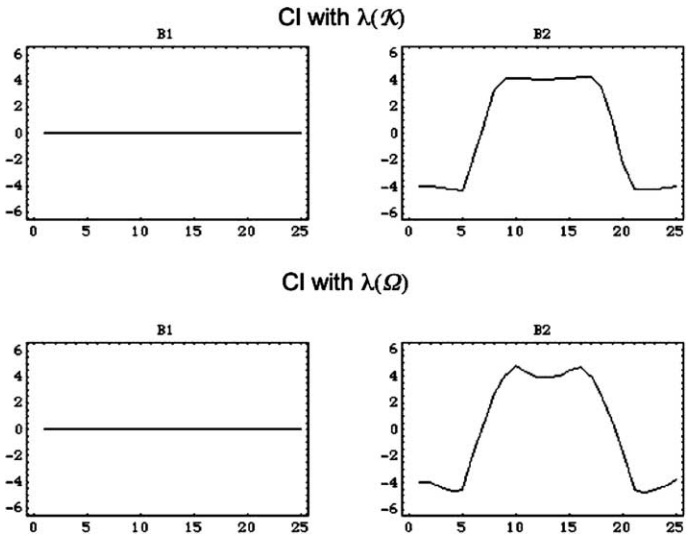

Fig. 9. Solution profiles on $\mathscr{T}_h^{100}$ for the tensor product mesh sequence using multiple (top) and single (bottom) parameter optimization for λ.

adjust contributions from low and high order potentials locally, it maintains an almost monotone profile of the discontinuous component of **B**.

From these experiments we can draw a conclusion that a single parameter strategy is appropriate for the remapping of smooth vector fields, while multiple parameter optimization should be used for discontinuous or rapidly changing fields.

Our next experiment compares CI and CT algorithms for the smooth field (24) and the two mesh sequences. The low order versions of each remapper are applied first. The energy plots of the Donor cell CT remapper; see [9] and the low order CI (with $\lambda = 1$) are shown in Fig. 10. On the random sequence the two remappers are virtually indistinguishable. For the tensor product sequence the low order CI is slightly more dissipative. Fig. 11 shows a further comparison of the remappers on the tensor product mesh sequence. In addition to the low order cases now we include data for the CT remapper with the monotone and Van Leer limiters, see [9,21], and data for the CI algorithm with a feedback loop control. We see that (18) provides for an almost constant energy level in the CI field. In contrast, the high order accuracy of the CT remapper is lost during the "compression" phases when the mesh rapidly moves from left to right and the cells near the right wall experience rapid change in their aspect ratios.

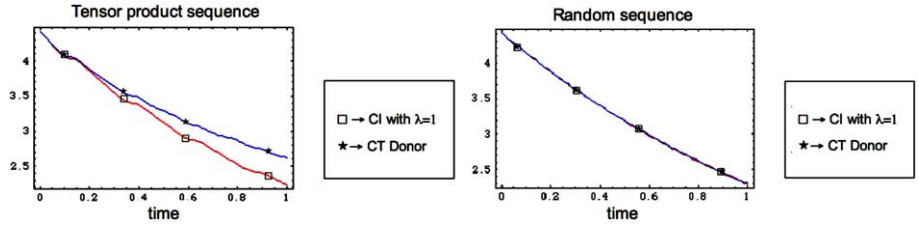

Fig. 10. Energy of the remapped solution under low order CI and CT remappers.

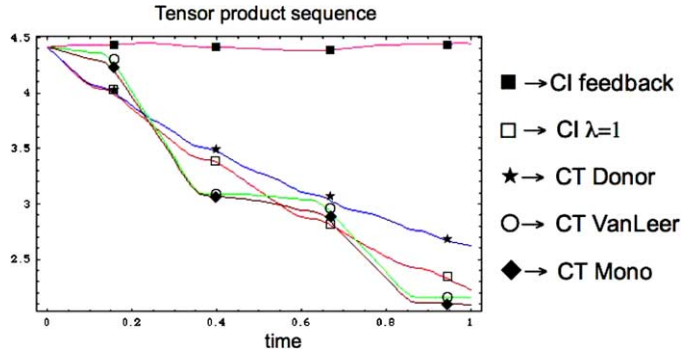

Fig. 11. Energy of the remapped solution. CI with feedback control and $\lambda = 1$. CT with donor cell, Van Leer and Monotone limiters.

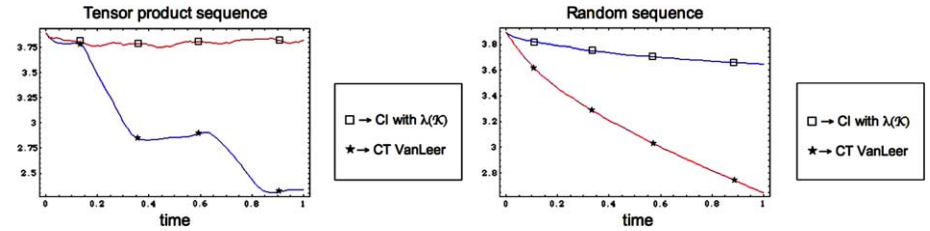

Fig. 12. Energy of (25). CI with multiple parameter optimization vs. CT with van Leer limiting.

The last experiment compares CI and CT remappers for the discontinuous field (25). In this case CI remap is applied with multiple parameter optimization for A. In the CT remapper, the Van Leer limiter is used. The associated energy plots of the remapped fields for the tensor product sequence are shown at the top in Fig. 12. The step-like energy profile under the CT remap continues to persists for this mesh sequence. The multiple parameter optimization strategy in CI is able to maintain the energy at about the same level. Fig. 13 shows the profiles of the solution at the final mesh. The energy loss under the CT remapper is clearly visible in the second component of (25) where discontinuity is completely smeared. In contrast, CI algorithm is able to maintain the crisp profile of the second component.

Energy plots for the random mesh sequence are shown in Fig. 12. In this case, the energy plot of the CT remapper with Van Leer limiting is indistinguishable from the one obtained with the simple Donor cell upwinding. This behavior is caused by the inability of the dimension by dimension reconstruction to provide a true high order interpolation when mesh faces are not approximately aligned with the coordinate axes. The CI method also tends to be more dissipative on the random mesh sequence. However, as

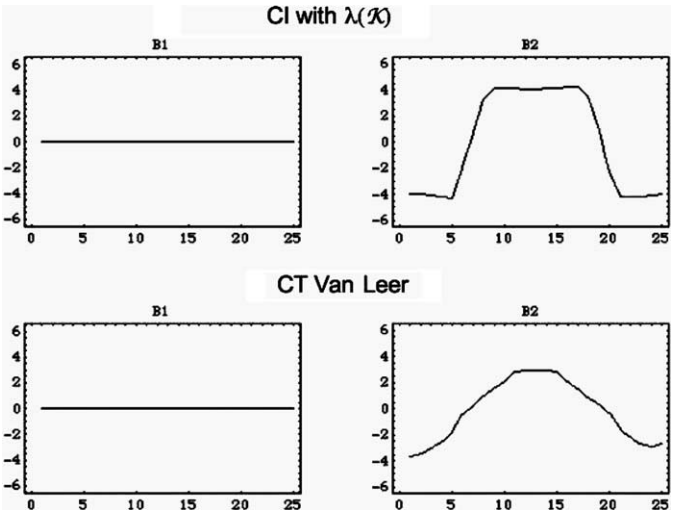

Fig. 13. Final profile of (25). CI with multiple parameter optimization vs. CT with van Leer limiting for tensor mesh sequence.

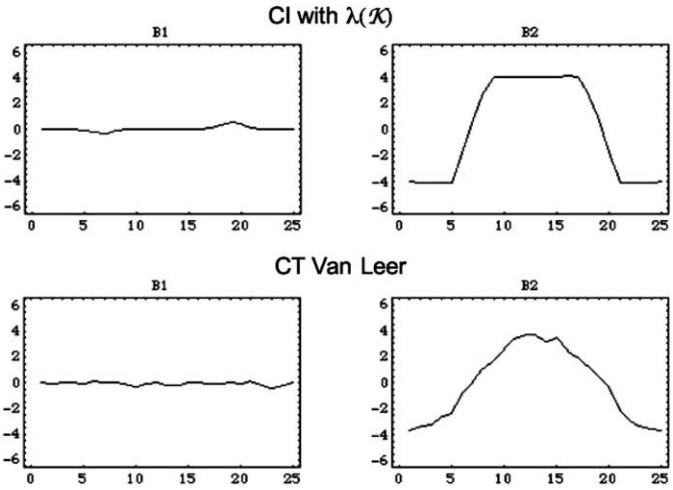

Fig. 14. Final profile of (25). CI with multiple parameter optimization vs. CT with van Leer limiting for random sequence.

Fig. 12 shows, its performance is substantially better than that of the CT remapper. This conclusion can be also confirmed by inspecting plots of solution profiles on the last mesh in the sequence, shown in Fig. 14. The discontinuity smearing under CT is clearly visible, while CI is capable of providing relatively sharp profiles of the second component in (25).

6. Conclusions

We have formulated a constrained interpolation (remap) algorithm for divergence-free vector fields in two dimensions. The use of interpolation instead of advection makes the new algorithm applicable to a

much broader range of settings than traditional advection based remappers. The CI algorithm can also be easily adapted to different discretizations, including finite difference and finite volume methods, provided a discrete exact sequence of spaces is available. Implementation of the algorithm is facilitated by its modular design that allows for an efficient incorporation of various postprocessing techniques for the vector potential. Numerical experiments demonstrate excellent performance of the new remapper, in particular, ability to maintain almost constant energy levels throughout the remap cycle and ability to maintain good resolution of sharp features. Extensions to three dimensions will be reported in a forthcoming paper.

Acknowledgments

We wish to thank A. Robinson for helpful discussions about remap problems and many valuable suggestions about the numerical tests in this paper. We also thank the anonymous referee for the careful reading of the paper and the detailed report that helped to improve the manuscript.

References

[1] R. Abgrall, P.L. Roe, High order fluctuations schemes on triangular meshes, J. Sci. Comput. 19 (1–2) (2003) 3–36.
[2] D. Balsara, Divergence-free adaptive mesh refinement for MHD, J. Comput. Phys. 174 (2001) 614–648.
[3] J.U. Brackbill, D.C. Barnes, The effects of nonzero $\nabla \cdot \mathbf{B}$ on the numerical solution of the magneto-hydrodynamic equations, J. Comput. Phys. 35/3 (1980) 426–430.
[4] H. De Sterck, Multi-dimensional upwind constrained transport on unstructured grids for Shallow Water MED, AIAA Paper, 2001, pp. 2001–2623.
[5] J. Dukowicz, J. Baumgardner, Incremental remapping as a transport/advection algorithm, J. Comput. Phys. 160 (2000) 318–335.
[6] P. Bochev, A. Robinson, Matching algorithms with physics: exact sequences of finite element spaces, in: D. Estep, S. Tavener (Eds.), Collected Lectures on the Preservation of Stability Under Discretization, SIAM, Philadelphia, 2001.
[7] D. Braess, Finite Elements: Theory, Fast Solvers, and Applications in Solid Mechanics, Cambridge University Press, Cambridge, UK, 1997.
[8] G. Carey, G. Bicken, V. Carey, C. Berger, J. Sanchez, Locally constrained projections on grids, Int. J. Numer. Methods Engrg. 50 (2001) 549–577.
[9] C.R. Evans, J.F. Hawley, Simulation of magnetohydrodynamic flows: a constrained transport method, Astrophys. J. 332 (1988) 659–677.
[10] V. Girault, L. Ridgway Scott, A quasi-local interpolation operator preserving the discrete divergence, Calcolo 40 (2003) 1–19.
[11] C. Hirt, A. Amsden, J. Cook, An arbitrary Lagrangian–Eulerian computing method for all flow speeds, J. Comput. Phys. 14 (1974) 227–253.
[12] J.M. Hyman, M. Shashkov, Mimetic discretizations for Maxwell's equations, J. Comput. Phys. 151 (1999) 881–909.
[13] J.M. Hyman, M. Shashkov, Natural discretizations for the divergence gradient and curl on logically rectangular grids, Int. J. Comput. Math Applic. 33 (1997) 88–104.
[14] J.M. Hyman, M. Shashkov, Adjoint operators for the natural discretizations of the divergence gradient and curl on logically rectangular grids, Appl. Numer. Math. 25 (1997) 413–442.
[15] S. Li, H. Li, A novel approach of maintaining divergence-free for adaptive mesh refinement, J. Computat. Phys. 199 (1) (2004) 1–15.
[16] T. Lin, H. Wang, Recovering the gradients of the solutions of second-order hyperbolic equations by interpolating the finite element solution, J. Comput. Acoust. 3/1 (1995) 27–56.
[17] T. Lin, H. Wong, A class of error estimators based on interpolating the finite element solutions for reaction–diffusion equations, IMA Vol. Math. Applic. 75 (1995) 129–152.
[18] R. Magesh, R. Ruhle, Quadratic reconstruction on arbitrary polygonal grids for 2nd-order conservation laws, in: R. Herbin, D. Kroner (Eds.), Finite Volumes for Complex Applications III, Kogan Page Science, London, Sterling, 2003.
[19] L.G. Margolin, M. Shashkov, Second-order sign-preserving conservative interpolation (remapping) on general grids, J. Comput. Phys. 184 (2003) 266–298.
[20] P.W. Rambo, Vector potential remap for 2D MHD, Lawrence Livermore National Laboratory Report UCRL-ID-132123, October 1998.

[21] A.C. Robinson, P.B. Bochev, P.W. Rambo; Constrained transport remap on unstructured quadrilateral and hexahedral grids Technical Report SAND2001-2146P, Sandia National Laboratories, July 2001. Available from <http://infoserve.sandia.gov/sand_doc/2001/012146p.pdf>.
[22] G. Toth, P.L. Roe, Divergence and curl-preserving prolongation and restriction formulas, J. Comput. Phys. 180 (2002) 736–750.
[23] G. Toth, The $\nabla \cdot B = 0$ constraint in shock-capturing magnetohydrodynamic codes, J. Comput. Phys. 161 (2000) 605–652.
[24] J.S. Van Welij, Calculation of eddy currents in terms of H on hexahedra, IEEE Trans. Magnet. 21 (1985) 2239–2241.
[25] Z. Zhang, A. Naga, Validation of the a posteriori error estimator based on polynomial preserving recovery for linear elements, Int. J. Numer. Methods Engrg., accepted.
[26] J. Xu, Z. Zhang, Analysis of recovery type A a posteriori error estimators for mildly structured grids, Math. Computat. 73 (2004) 1139–1152.