Two arbitrary-order constraint-preserving schemes for the Yang–Mills equations on polyhedral meshes

Jérôme Droniou, Jia Jia Qian

https://www.aimspress.com/journal/mine

Mathematics in Engineering, 6(3): 468–493.
DOI:10.3934/mine.2024019
Received: 15 June 2023
Revised: 10 May 2024
Accepted: 17 June 2024
Published: 24 June 2024

Research article

Two arbitrary-order constraint-preserving schemes for the Yang–Mills equations on polyhedral meshes†

Jérôme Droniou[1,2,*] **and Jia Jia Qian**[2]

[1] IMAG, Univ. Montpellier, CNRS, Montpellier, France

[2] School of Mathematics, Monash University, Melbourne, Australia; jerome.droniou@monash.edu, jia.qian@monash.edu

† **This contribution is part of the Special Issue:** Advancements in Polytopal Element Methods
Guest Editors: Michele Botti; Franco Dassi; Lorenzo Mascotto; Ilario Mazzieri
Link: www.aimspress.com/mine/article/6538/special-articles

* **Correspondence:** Email: jerome.droniou@umontpellier.fr.

Abstract: Two numerical schemes are proposed and investigated for the Yang–Mills equations, which can be seen as a nonlinear generalisation of the Maxwell equations set on Lie algebra-valued functions, with similarities to certain formulations of General Relativity. Both schemes are built on the Discrete de Rham (DDR) method, and inherit from its main features: an arbitrary order of accuracy, and applicability to generic polyhedral meshes. They make use of the complex property of the DDR, together with a Lagrange-multiplier approach, to preserve, at the discrete level, a nonlinear constraint associated with the Yang–Mills equations. We also show that the schemes satisfy a discrete energy dissipation (the dissipation coming solely from the implicit time stepping). Issues around the practical implementations of the schemes are discussed; in particular, the assembly of the local contributions in a way that minimises the price we pay in dealing with nonlinear terms, in conjunction with the tensorisation coming from the Lie algebra. Numerical tests are provided using a manufactured solution, and show that both schemes display a convergence in L^2-norm of the potential and electrical fields in $O(h^{k+1})$ (provided that the time step is of that order), where k is the polynomial degree chosen for the DDR complex. We also numerically demonstrate the preservation of the constraint.

Keywords: discrete de Rham method; Yang–Mills equations; polytopal method; constraint-preserving scheme; energy estimate; 3D numerical tests

1. Introduction

In this paper we investigate two arbitrary-order numerical methods for the Yang–Mills equations on general polyhedral meshes, based on the fully discrete serendipity Discrete de Rham (SDDR) complex [1]. The first method was proposed (but not tested) in [2] for the non-serendipity version of the Discrete de Rham (DDR) sequence, while the second method is novel to this paper. The two discretisations differ in the treatment of one of the nonlinearities present in these equations. In contrast to conforming methods, the discrete structure of the SDDR spaces means that there is no obvious construction of the nonlinear terms, and this can be problematic when specific algebraic manipulations need to be reproduced, for instance to prove consistency estimates. The implementation cost is another important factor to the viability of each approach, which is explored with accompanying numerical results on the convergence and discrete conservation properties of each scheme.

The classical Yang–Mills equations come from a class of non-abelian gauge theories, generalising the abelian $U(1)$ group of electromagnetism to certain non-abelian gauge groups. Once quantised, this theory forms the foundation of the current Standard Model of particle physics. In the classical setting, the non-commutativity of the group manifests as the appearance of nonlinear quantities in addition to the linear Maxwell terms. Analogous to Maxwell, the Yang–Mills equations can be formulated as a set of evolution equations preserving particular constraints (e.g., the conservation of charge), given that the initial data satisfies these constraints. In the linear case, the preservation of these constraints is a direct consequence of the calculus formula $\operatorname{div}\mathbf{curl} = 0$, which is linked to the complex property of the de Rham sequence. Designing numerical methods that replicate this property is essential to maintaining constraint preservation at the discrete level, and thus to obtaining stable schemes. Much work has been done in the Finite Element framework to design discrete versions of the de Rham complex, see, e.g., [3–8] and references therein. Finite Element methods are, however, limited to meshes made of specific elements (mostly tetrahedra and hexahedra in 3D), which limits their flexibility in terms of mesh refinement or agglomeration. Recently, discrete polytopal complexes – discrete versions of continuous complexes, that are applicable on meshes made of generic polyhedra – have been introduced, see, e.g., [9–12]. The discrete complex property enabled the design of stable and robust schemes, in particular for magnetostatics [10, 13], plate problems [14–17], and the Stokes equations [18, 19]).

Given the importance, for the stability of schemes, of preserving constraints at the discrete level, similar techniques have been explored for the Yang–Mills equations, using either Finite Element or polytopal approaches [2, 20, 21]. For these equations, however, the nonlinearity has proven to be troublesome, and required additional techniques (e.g., the introduction of Lagrange multipliers) beyond a discrete version of the formula $\operatorname{div}\mathbf{curl} = 0$. The interest in developing our understanding of such methods is in the application to numerical schemes for Einstein's equations, where the absence of this constraint propagation can cause disastrous error growth [22–24] in the numerical simulations. Current techniques to control this error include constraint damping [22, 23, 25], where specific terms are added to the evolution equations to suppress the growth of the constraint violations, but methods for exact preservation remain limited. The link with the Yang–Mills equations is that in certain formulations of General Relativity (GR), such as the Einstein-Bianchi system [26, 27], these equations can resemble greatly those of electromagnetism with additional nonlinear terms. Therefore it is natural to expect that these ideas will aid in designing a constraint preserving scheme for GR based on the framework of discrete polytopal complexes.

An equally important aspect of the design of numerical methods is the feasibility of the implementation and testing under real world conditions. We find more commonly, for the Yang–Mills equations, numerical tests run in only low-order 2D settings [20, 21]. Any increase in the dimension or the order of the approximation generally leads to schemes that are vastly more expensive to run, and this is compounded by the nonlinearity of the model. Hence working with spaces that are smaller and more refined is an effective way to cut the cost of the simulations. The SDDR complex, introduced in [1], is a variant of the DDR complex [11, 12], where the spaces have undergone a serendipity reduction, eliminating many unknowns, while retaining the complex and consistency properties of the original sequence. This enables the seamless transfer of any DDR scheme and results to the SDDR version, with all the flexibility of the general-order polytopal method at a lower cost. Additionally, this can be combined with other reduction techniques such as static condensation to further increase the efficiency.

The issue with the nonlinearity in the practical implementation is the computations involving 'high dimensional' arrays that are required to deal with all the coefficients. This number grows exponentially with the degree of the multilinearity, and thus takes up majority of the time in the assembly phase of the runtime. For matrices (2-dimensional arrays), there exists many specialised algorithms to speed up calculations, as well as efficient storage structures in the case that it is sparse. The libraries for higher dimensional arrays are less advanced, and often incomplete in their features; as a consequence, operations need to be done manually, introducing another source of possible inefficiencies. Simply rearranging the order of calculations can lead to sizeable differences in the space and time complexities, therefore finding the optimal trade-off is key to measuring the actual performance of the scheme.

The paper is organised as follows: In Section 2, we give a presentation of the SDDR complex and its Lie algebra extension, that is independent of the DDR framework. Section 3 starts with the constrained formulation of the continuous Yang–Mills equations. Based on that, we introduce the two schemes that are considered in the paper and the differing approaches on the nonlinear terms. This is followed by a proof of the preservation of a discrete constraint functional, as well as energy estimates. Section 4 covers the major steps in the implementation of the scheme, showing the impact of Lie algebra tensorisation on the physical data structures, and also highlighting how the trilinear and quadrilinear forms and sums are managed in the tensorised situation. Numerical results for these implementations are found in Section 5, where we test both the convergence and the discrete constraint preservation on three different mesh families in the 3D setting. Expected convergence rates of $k + 1$ are mostly seen, as well as the preservation of the initial constraint up to machine precision. We also report on the differences in the results and runtimes, which turned out to be very minor between the two methods. A brief conclusion is provided in Section 6.

2. Lie Algebra-valued serendipity Discrete de Rham complex

We present here the serendipity version of the arbitrary-order Lie Algebra-valued DDR complex, originally sketched in [2, Section 6]. This complex consists in tensorising the SDDR complex of [1], which is built in this reference from the (regular) DDR complex; such a presentation relies on a complete description of the latter complex, together with "extension" and "reduction" maps that link the two complexes. In the following, we adopt a stand-alone description of the SDDR complex, directly translating the formulas resulting from the links with the DDR complex. For this reason, the notations adopted below differ slightly from [1]: there, the serendipity spaces and operators are denoted using a

hat (the non-hat version referring to the regular DDR spaces and operators, which are not needed here).

2.1. Mesh notations

We use the same mesh and polynomial space notations as in [12]. Let U be a polygonal domain of $\mathbb{R}^3$. A mesh $\mathcal{M}_h = \mathcal{T}_h \cup \mathcal{F}_h \cup \mathcal{E}_h \cup \mathcal{V}_h$ is a collection of polyhedral elements (gathered in $\mathcal{T}_h$, and partitioning U), of polygonal faces (gathered in $\mathcal{F}_h$), of edges (gathered in $\mathcal{E}_h$) and vertices (gathered in $\mathcal{V}_h$). Each $\mathsf{P} \in \mathcal{M}_h$ is assumed to be topologically trivial (simply connected with connected boundary), and we denote by h_P the diameter of P; we set $h = \max_{T\in\mathcal{T}_h} h_T$. When applicable, the sets $\mathcal{F}_\mathsf{P}$ (resp. $\mathcal{E}_\mathsf{P}$, resp. $\mathcal{V}_\mathsf{P}$) gather the faces (resp. edges, resp. vertices) of P. Each face $F \in \mathcal{F}_h$ is oriented by the choice of a unit normal $\boldsymbol{n}_F$, and each edge $E \in \mathcal{E}_h$ is oriented by the choice of a unit tangent $\boldsymbol{t}_E$. If $T \in \mathcal{T}_h$ and $F \in \mathcal{F}_T$, ω_{TF} is the relative orientation of F with respect to T: $\omega_{TF} = +1$ if $\boldsymbol{n}_F$ points outside T, $\omega_{TF} = -1$ otherwise. For $F \in \mathcal{F}_h$ and $E \in \mathcal{E}_F$, ω_{FE} denotes the relative orientation of E with respect to F: $\omega_{FE} = +1$ if, along ∂F, $\boldsymbol{t}_E$ points counter-clockwise with respect to the orientation of F induced by $\boldsymbol{n}_F$, and $\omega_{FE} = -1$ otherwise; we also denote by $\boldsymbol{n}_{FE}$ the unit vector such that $(\boldsymbol{t}_E, \boldsymbol{n}_{FE}, \boldsymbol{n}_F)$ defines a right-handed system in $\mathbb{R}^3$. The final orientation is that of the vertices of each edge: for $E \in \mathcal{E}_h$ and $V \in \mathcal{V}_E$, $\omega_{EV} = +1$ if $\boldsymbol{t}_E$ points towards V on E, and $\omega_{EV} = -1$ otherwise.

We assume that $\mathcal{T}_h \cup \mathcal{F}_h$ satisfies the regularity assumption of [28, Definition 1.9] with regularity parameter ϱ, and we write $A \lesssim B$ when $A \le CB$ for some C depending only on U, ϱ and the possible polynomial degrees involved in A, B.

For any mesh face $F \in \mathcal{F}_h$ and smooth enough function $r : F \to \mathbb{R}$, $\mathbf{grad}_F\, r$ is the gradient of r on F and $\mathbf{rot}_F\, r$ its 2-dimensional vector curl (rotation of $\mathbf{grad}_F\, r$ by $-\pi/2$ in the plane spanned by F). For a smooth function $\boldsymbol{z}$ on F with values in the tangent plane of F, the divergence of $\boldsymbol{z}$ on F is $\mathrm{div}_F\, \boldsymbol{z}$, and its scalar curl (divergence of the rotated by $-\pi/2$ of $\boldsymbol{z}$) is $\mathrm{rot}_F\, \boldsymbol{z}$.

If $\mathsf{P} \in \mathcal{M}_h$ and $\ell \ge 0$ is an integer, $\mathcal{P}^\ell(\mathsf{P})$ denotes the space of restrictions to P of three-variate polynomials on $\mathbb{R}^3$ of total degree $\le \ell$, and $\mathcal{P}^{0,\ell}(\mathsf{P})$ is its subspace of polynomials with vanishing integral over P. We adopt the convention $\mathcal{P}^\ell(\mathsf{P}) = \{0\}$ if $\ell < 0$. If $T \in \mathcal{T}_h$, we set $\boldsymbol{\mathcal{P}}^\ell(T) = \mathcal{P}^\ell(T)^3$ and, for $F \in \mathcal{F}_h$, $\boldsymbol{\mathcal{P}}^\ell(F)$ is the subspace of $\mathcal{P}^\ell(F)^3$ which take value in the tangent space of F. The L^2-orthogonal projector on $\mathcal{P}^\ell(\mathsf{P})$ is denoted by $\pi^\ell_{\mathcal{P},\mathsf{P}}$. Selecting, for each $\mathsf{P} \in \mathcal{T}_h \cup \mathcal{F}_h$, a point $\boldsymbol{x}_\mathsf{P} \in \mathsf{P}$ such that P contains a ball centered at $\boldsymbol{x}_\mathsf{P}$ and of radius $\gtrsim h_\mathsf{P}$, we recall the following decompositions of vector-valued polynomial spaces: For all $F \in \mathcal{F}_h$,

$$\boldsymbol{\mathcal{P}}^\ell(F) = \boldsymbol{\mathcal{R}}^\ell(F) \oplus \boldsymbol{\mathcal{R}}^{\mathrm{c},\ell}(F) \quad \text{with} \quad \boldsymbol{\mathcal{R}}^\ell(F) = \mathbf{rot}_F\, \mathcal{P}^{\ell+1}(F) \text{ and } \boldsymbol{\mathcal{R}}^{\mathrm{c},\ell}(F) = (\boldsymbol{x} - \boldsymbol{x}_F)\mathcal{P}^{\ell-1}(F)$$

and, for $T \in \mathcal{T}_h$,

$$\boldsymbol{\mathcal{P}}^\ell(T) = \boldsymbol{\mathcal{R}}^\ell(T) \oplus \boldsymbol{\mathcal{R}}^{\mathrm{c},\ell}(T) \quad \text{with} \quad \boldsymbol{\mathcal{R}}^\ell(T) = \mathbf{curl}\, \boldsymbol{\mathcal{P}}^{\ell+1}(T) \text{ and } \boldsymbol{\mathcal{R}}^{\mathrm{c},\ell}(T) = (\boldsymbol{x} - \boldsymbol{x}_T)\mathcal{P}^{\ell-1}(T),$$
$$\boldsymbol{\mathcal{P}}^\ell(T) = \boldsymbol{\mathcal{G}}^\ell(T) \oplus \boldsymbol{\mathcal{G}}^{\mathrm{c},\ell}(T) \quad \text{with} \quad \boldsymbol{\mathcal{G}}^\ell(T) = \mathbf{grad}\, \mathcal{P}^{\ell+1}(T) \text{ and } \boldsymbol{\mathcal{G}}^{\mathrm{c},\ell}(T) = (\boldsymbol{x} - \boldsymbol{x}_T) \times \boldsymbol{\mathcal{P}}^{\ell-1}(T),$$

(here and in the following, when used between two vectors or a vector and a space, $\times$ denotes the cross product in $\mathbb{R}^3$). The L^2-orthogonal projectors on these spaces are, with obvious notations, $\boldsymbol{\pi}^\ell_{\mathcal{R},F}$, $\boldsymbol{\pi}^{\mathrm{c},\ell}_{\mathcal{R},F}$, $\boldsymbol{\pi}^\ell_{\mathcal{R},T}$, $\boldsymbol{\pi}^{\mathrm{c},\ell}_{\mathcal{R},T}$, $\boldsymbol{\pi}^\ell_{\mathcal{G},T}$ and $\boldsymbol{\pi}^{\mathrm{c},\ell}_{\mathcal{G},T}$.

2.2. Serendipity DDR complex

For each element or face $\mathsf{P} \in \mathcal{T}_h \cup \mathcal{F}_h$, we select on the boundary of the element (resp. face) a set $\mathcal{B}_\mathsf{P}$ of $\eta_\mathsf{P} \ge 2$ faces (resp. edges) that are not pairwise coplanar (resp. aligned) and such that, for each

$\mathsf{b} \in \mathcal{B}_\mathsf{P}$, P lies entirely on one side of the affine space spanned by b. From here on, we fix a polynomial degree $k \geq 0$, measuring the accuracy of the discrete complex, and we set

$$\ell_\mathsf{P} = k + 1 - \eta_\mathsf{P}, \qquad \forall \mathsf{P} \in \mathcal{T}_h \cup \mathcal{F}_h.$$

2.2.1. Spaces and serendipity operators

The SDDR versions of the $H^1(U)$, $\boldsymbol{H}(\mathbf{curl}; U)$, $\boldsymbol{H}(\mathrm{div}; U)$ and $L^2(U)$ spaces appearing in the continuous de Rham complex are the following spaces.

$$\begin{aligned}
\underline{X}^k_{\mathbf{grad},h} &:= \Big\{\underline{q}_h = ((q_T)_{T\in\mathcal{T}_h}, (q_F)_{F\in\mathcal{F}_h}, (q_E)_{E\in\mathcal{E}_h}, (q_V)_{V\in\mathcal{V}_h}) \ : \\
&\qquad q_T \in \mathcal{P}^{\ell_T}(T) \text{ for all } T \in \mathcal{T}_h,\, q_F \in \mathcal{P}^{\ell_F}(F) \text{ for all } F \in \mathcal{F}_h, \\
&\qquad q_E \in \mathcal{P}^{k-1}(E) \text{ for all } E \in \mathcal{E}_h, \text{ and } q_V \in \mathbb{R} \text{ for all } V \in \mathcal{V}_h\Big\}, \\
\underline{\boldsymbol{X}}^k_{\mathbf{curl},h} &:= \Big\{\underline{\boldsymbol{v}}_h = ((\boldsymbol{v}_{\mathcal{R},T}, \boldsymbol{v}^{\mathrm{c}}_{\mathcal{R},T})_{T\in\mathcal{T}_h}, (\boldsymbol{v}_{\mathcal{R},F}, \boldsymbol{v}^{\mathrm{c}}_{\mathcal{R},F})_{F\in\mathcal{F}_h}, (v_E)_{E\in\mathcal{E}_h}) \ : \\
&\qquad \boldsymbol{v}_{\mathcal{R},T} \in \boldsymbol{\mathcal{R}}^{k-1}(T) \text{ and } \boldsymbol{v}^{\mathrm{c}}_{\mathcal{R},T} \in \boldsymbol{\mathcal{R}}^{\mathrm{c},\ell_T+1}(T) \text{ for all } T \in \mathcal{T}_h, \\
&\qquad \boldsymbol{v}_{\mathcal{R},F} \in \boldsymbol{\mathcal{R}}^{k-1}(F) \text{ and } \boldsymbol{v}^{\mathrm{c}}_{\mathcal{R},F} \in \boldsymbol{\mathcal{R}}^{\mathrm{c},\ell_F+1}(F) \text{ for all } F \in \mathcal{F}_h, \\
&\qquad \text{and } v_E \in \mathcal{P}^k(E) \text{ for all } E \in \mathcal{E}_h\Big\}, \\
\underline{\boldsymbol{X}}^k_{\mathrm{div},h} &:= \Big\{\underline{\boldsymbol{w}}_h = ((\boldsymbol{w}_{\mathcal{G},T}, \boldsymbol{w}^{\mathrm{c}}_{\mathcal{G},T})_{T\in\mathcal{T}_h}, (w_F)_{F\in\mathcal{F}_h}) \ : \\
&\qquad \boldsymbol{w}_{\mathcal{G},T} \in \boldsymbol{\mathcal{G}}^{k-1}(T) \text{ and } \boldsymbol{w}^{\mathrm{c}}_{\mathcal{G},T} \in \boldsymbol{\mathcal{G}}^{\mathrm{c},k}(T) \text{ for all } T \in \mathcal{T}_h, \\
&\qquad \text{and } v_F \in \mathcal{P}^k(F) \text{ for all } F \in \mathcal{F}_h\Big\}, \\
\mathcal{P}^k(\mathcal{T}_h) &:= \Big\{r_h \in L^2(U) \ : \ (r_h)_{|T} \in \mathcal{P}^k(T) \text{ for all } T \in \mathcal{T}_h\Big\}.
\end{aligned}$$

The interpolators on these spaces consist in projecting continuous scalar/vector fields (or some of their traces) onto the polynomial components of the spaces. Specifically,

$$\underline{I}^k_{\mathbf{grad},h} : C(\overline{U}) \to \underline{X}^k_{\mathbf{grad},h},$$

$$\underline{\boldsymbol{I}}^k_{\mathbf{curl},h} : \boldsymbol{C}(\overline{U}) \to \underline{\boldsymbol{X}}^k_{\mathbf{curl},h}$$

and

$$\underline{\boldsymbol{I}}^k_{\mathrm{div},h} : \boldsymbol{C}(\overline{U}) \to \underline{\boldsymbol{X}}^k_{\mathrm{div},h}$$

are defined as:

$$\begin{aligned}
\underline{I}^k_{\mathbf{grad},h} q &= ((\pi^{\ell_T}_{\mathcal{P},T} q)_{T\in\mathcal{T}_h}, (\pi^{\ell_F}_{\mathcal{P},F} q)_{F\in\mathcal{F}_h}, (\pi^{k-1}_{\mathcal{P},E} q)_{E\in\mathcal{E}_h}, (q(\boldsymbol{x}_V))_{V\in\mathcal{V}_h}), && \forall q \in C(\overline{U}), \\
\underline{\boldsymbol{I}}^k_{\mathbf{curl},h} \boldsymbol{v} &= ((\boldsymbol{\pi}^{k-1}_{\mathcal{R},T} \boldsymbol{v}, \boldsymbol{\pi}^{\mathrm{c},\ell_T+1}_{\mathcal{R},T} \boldsymbol{v})_{T\in\mathcal{T}_h}, (\boldsymbol{\pi}^{k-1}_{\mathcal{R},F} \boldsymbol{v}_{\mathrm{t},F}, \boldsymbol{\pi}^{\mathrm{c},\ell_F+1}_{\mathcal{R},F} \boldsymbol{v}_{\mathrm{t},F})_{F\in\mathcal{F}_h}, (\pi^k_{\mathcal{P},E}(\boldsymbol{v}\cdot\boldsymbol{t}_E))_{E\in\mathcal{E}_h}), && \forall \boldsymbol{v} \in \boldsymbol{C}(\overline{U}), \\
\underline{\boldsymbol{I}}^k_{\mathrm{div},h} \boldsymbol{w} &= ((\boldsymbol{\pi}^{k-1}_{\mathcal{G},T} \boldsymbol{w}, \boldsymbol{\pi}^{\mathrm{c},k}_{\mathcal{G},T} \boldsymbol{w})_{T\in\mathcal{T}_h}, (\pi^k_{\mathcal{P},F}(\boldsymbol{w}\cdot\boldsymbol{n}_E))_{E\in\mathcal{E}_h}), && \forall \boldsymbol{w} \in \boldsymbol{C}(\overline{U}),
\end{aligned}$$

where $\boldsymbol{v}_{\mathrm{t},F} = \boldsymbol{n}_F \times (\boldsymbol{v}_{|F} \times \boldsymbol{n}_F)$ is the tangential trace of $\boldsymbol{v}$ on F.

As usual in fully discrete complexes, we adopt the underlined notation for vectors of polynomial components, and we replace the index h with P to denote the restriction of these spaces (and the

operators defined on them) to a mesh entity $\mathsf{P} \in \mathcal{T}_h \cup \mathcal{F}_h \cup \mathcal{E}_h$ and its boundary entities. So, for example, a vector $\underline{\boldsymbol{v}}_F \in \underline{\boldsymbol{X}}^k_{\mathbf{curl},F}$ corresponds to $\underline{\boldsymbol{v}}_F = (\boldsymbol{v}_{\mathcal{R},F}, \boldsymbol{v}^{\mathrm{c}}_{\mathcal{R},F}, (v_E)_{E\in\mathcal{E}_F})$.

The DDR spaces correspond to the spaces above with the choice $\ell_F = \ell_T = k-1$ (that is, $\eta_F = \eta_P = 2$). This implies in particular that, for $k = 0$ (which forces $\ell_\mathsf{P} < 0$ for all $\mathsf{P} \in \mathcal{T}_h \cup \mathcal{F}_h$), the standard and serendipity DDR spaces are identical. However, as soon as $k \geq 1$, the SDDR spaces have lower dimensions, while still encoding the same level of polynomial consistency as the DDR spaces. This is due to the existence of two families of key operators, the serendipity gradient and curl operators. Specifically, for $\mathsf{P} \in \mathcal{T}_h \cup \mathcal{F}_h$, the role of the gradient serendipity operator $\boldsymbol{S}^k_{\mathbf{grad},\mathsf{P}} : \underline{X}^k_{\mathbf{grad},\mathsf{P}} \to \boldsymbol{\mathcal{P}}^k(\mathsf{P})$ is to reconstruct a consistent gradient, while the curl serendipity operator $\boldsymbol{S}^k_{\mathbf{curl},\mathsf{P}} : \underline{\boldsymbol{X}}^k_{\mathbf{curl},\mathsf{P}} \to \boldsymbol{\mathcal{P}}^k(\mathsf{P})$ reconstructs a consistent vector potential. The consistencies in questions are expressed by the following relations (see [1, Proposition 18]):

$$\boldsymbol{S}^k_{\mathbf{grad},\mathsf{P}} \underline{I}^k_{\mathbf{grad},\mathsf{P}} q = \mathbf{grad}_\mathsf{P}\, q, \qquad \forall q \in \mathcal{P}^{k+1}(\mathsf{P})$$

$$\boldsymbol{S}^k_{\mathbf{curl},\mathsf{P}} \underline{\boldsymbol{I}}^k_{\mathbf{curl},\mathsf{P}} \boldsymbol{v} = \boldsymbol{v}, \qquad \forall \boldsymbol{v} \in \boldsymbol{\mathcal{P}}^k(\mathsf{P}).$$

We do not present the precise definitions of these operators, which are not essential to describe the SDDR complex, and refer the reader to [1].

In the next three sections we define operators acting on these spaces, with values in full polynomial spaces, mimicking the gradient, curl, and divergence. It should be noted that, in the original presentation of the SDDR complex in [1], these operators were not explicitly defined – only the discrete operators (projections on the complex spaces, see Section 2.2.5) and discrete inner products were detailed, based on those of the DDR complex. The polynomial operators below correspond to those of the DDR complex composed with the extension operators linking the DDR and SDDR complex; for ease of reference, we indicate which formulas from [1] yield the definitions presented here.

2.2.2. Operators on the gradient space

For each edge $E \in \mathcal{E}_h$ we define the edge gradient $G^k_E : \underline{X}^k_{\mathbf{grad},E} \to \mathcal{P}^k(E)$ and potential reconstruction $\gamma^{k+1}_E : \underline{X}^k_{\mathbf{grad},E} \to \mathcal{P}^{k+1}(E)$ by: For all $\underline{q}_E = (q_E, (q_V)_{V\in\mathcal{V}_E}) \in \underline{X}^k_{\mathbf{grad},E}$,

$$\int_E G^k_E \underline{q}_E\, r_E = -\int_E q_E r'_E + \sum_{V\in\mathcal{V}_E} \omega_{EV} q_V r_E(\boldsymbol{x}_V), \qquad \forall r_E \in \mathcal{P}^k(E),$$

$$\gamma^{k+1}_E \underline{q}_E(\boldsymbol{x}_V) = q_V \quad \forall V \in \mathcal{V}_E \quad \text{and} \quad \pi^{k-1}_{\mathcal{P},E}(\gamma^{k+1}_E \underline{q}_E) = q_E.$$

The definition of $G^k_E \underline{q}_E$, in which the derivative r'_E is taken in the direction $\boldsymbol{t}_E$, mimics an integration-by-parts formula; it can be checked that

$$G^k_E \underline{q}_E = (\gamma^{k+1}_E \underline{q}_E)'.$$

For each $F \in \mathcal{F}_h$, combining [1, Eqs (4.2), (5.18) and (6.6)] together with $\mathrm{div}_F\, \mathbf{rot}_F = 0$ yields the following definition of the face gradient $\mathbf{G}^k_F : \underline{X}^k_{\mathbf{grad},F} \to \boldsymbol{\mathcal{P}}^k(F)$: For all $\underline{q}_F \in \underline{X}^k_{\mathbf{grad},F}$,

$$\int_F \mathbf{G}^k_F \underline{q}_F \cdot (\boldsymbol{w} + \boldsymbol{\tau}) = \sum_{E\in\mathcal{E}_F} \omega_{FE} \int_E \gamma^{k+1}_E \underline{q}_E (\boldsymbol{w}\cdot\boldsymbol{n}_{FE}) + \int_F \boldsymbol{S}^k_{\mathbf{grad},F} \underline{q}_F \cdot \boldsymbol{\tau}, \quad \forall (\boldsymbol{w},\boldsymbol{\tau}) \in \boldsymbol{\mathcal{R}}^k(F) \times \boldsymbol{\mathcal{R}}^{\mathrm{c},k}(F).$$

Using this face gradient and [1, Eqs (4.3) and (5.18)], the scalar potential reconstruction on $F \in \mathcal{F}_h$ is then $\gamma_F^{k+1} : \underline{X}^k_{\mathbf{grad},F} \to \mathcal{P}^{k+1}(F)$ defined by: For all $\underline{q}_F \in \underline{X}^k_{\mathbf{grad},F}$,

$$\int_F \gamma_F^{k+1}\underline{q}_F \operatorname{div}_F \boldsymbol{w} = -\int_F \mathbf{G}_F^k\underline{q}_F \cdot \boldsymbol{w} + \sum_{E\in\mathcal{E}_F} \omega_{FE}\int_E \gamma_E^{k+1}\underline{q}_E(\boldsymbol{w}\cdot\boldsymbol{n}_{FE}), \qquad \forall \boldsymbol{w} \in \boldsymbol{\mathcal{R}}^{\mathrm{c},k+2}(F).$$

Finally, for $T \in \mathcal{T}_h$, we use [1, Eqs (4.4), (5.32) and (6.6)] to write the element gradient $\mathbf{G}_T^k : \underline{X}^k_{\mathbf{grad},T} \to \boldsymbol{\mathcal{P}}^k(T)$ as: For all $\underline{q}_T \in \underline{X}^k_{\mathbf{grad},T}$,

$$\int_T \mathbf{G}_T^k\underline{q}_T \cdot (\boldsymbol{w}+\boldsymbol{\tau}) = \sum_{F\in\mathcal{F}_T} \omega_{TF}\int_F \gamma_F^{k+1}\underline{q}_F(\boldsymbol{w}\cdot\boldsymbol{n}_{TF}) + \int_T \boldsymbol{S}^k_{\mathbf{grad},T}\underline{q}_T\cdot\boldsymbol{\tau}, \quad \forall(\boldsymbol{w},\boldsymbol{\tau}) \in \boldsymbol{\mathcal{R}}^k(T)\times\boldsymbol{\mathcal{R}}^{\mathrm{c},k}(T).$$

The potential reconstruction $P^{k+1}_{\mathbf{grad},T} : \underline{X}^k_{\mathbf{grad},T} \to \mathcal{P}^{k+1}(T)$ is such that: For all $\underline{q}_T \in \underline{X}^k_{\mathbf{grad},T}$,

$$\int_T P^{k+1}_{\mathbf{grad},T}\underline{q}_T \operatorname{div}\boldsymbol{w} = -\int_T \mathbf{G}_T^k\underline{q}_T\cdot\boldsymbol{w} + \sum_{F\in\mathcal{F}_T}\omega_{TF}\int_F \gamma_F^{k+1}\underline{q}_F(\boldsymbol{w}\cdot\boldsymbol{n}_{TF}), \qquad \forall \boldsymbol{w} \in \boldsymbol{\mathcal{R}}^{\mathrm{c},k+2}(T).$$

Remark 1 (Approximation properties of the potential reconstructions in $\underline{X}^k_{\mathbf{grad},T}$). As demonstrated by [12, Theorem 6], the potential reconstructions on the space $\underline{X}^k_{\mathbf{grad},T}$ have optimal approximation properties of degree $k+1$. This is however an exception to the rule of spaces and potential reconstructions in the DDR complex; the reasons for this exception are better understood when translating this complex in the language of differential forms (see [29], especially Remarks 7 and 18 therein).

2.2.3. Operators on the curl space

For $F \in \mathcal{F}_h$, using [1, Eqs (4.6), (5.19) and (6.7)] we define the face curl $C_F^k : \underline{\boldsymbol{X}}^k_{\mathbf{curl},F} \to \mathcal{P}^k(F)$ by: For all $\underline{\boldsymbol{v}}_F \in \underline{\boldsymbol{X}}^k_{\mathbf{curl},F}$,

$$\int_F C_F^k\underline{\boldsymbol{v}}_F r = \int_F \boldsymbol{v}_{\mathcal{R},F}\cdot\mathbf{rot}_F\, r - \sum_{E\in\mathcal{E}_F}\omega_{FE}\int_E v_E r, \qquad \forall r \in \mathcal{P}^k(F).$$

This definition is actually identical to the face curl in the DDR complex and does not invoke $\boldsymbol{S}^k_{\mathbf{curl},F}$, as the extension operators between the SDDR and DDR curl face space do not modify the components on $\boldsymbol{\mathcal{R}}^{k-1}(F)$ and on $\bigtimes_{E\in\mathcal{E}_F}\mathcal{P}^k(E)$. The curl serendipity operator is however involved in the definition of the component on $\boldsymbol{\mathcal{R}}^{\mathrm{c},k}(F)$ of the extension (see [1, Eq (5.19)]), and therefore in the vector potential reconstruction $\boldsymbol{\gamma}^k_{\mathrm{t},F} : \underline{\boldsymbol{X}}^k_{\mathbf{curl},F} \to \boldsymbol{\mathcal{P}}^k(F)$ on F, defined as: For all $\underline{\boldsymbol{v}}_F \in \underline{\boldsymbol{X}}^k_{\mathbf{curl},F}$,

$$\int_F \boldsymbol{\gamma}^k_{\mathrm{t},F}\underline{\boldsymbol{v}}_F\cdot(\mathbf{rot}_F\, r+\boldsymbol{\tau}) = \int_F C_F^k\underline{\boldsymbol{v}}_F r + \sum_{E\in\mathcal{E}_F}\omega_{FE}\int_E v_E r + \int_F \boldsymbol{S}^k_{\mathbf{curl},F}\underline{\boldsymbol{v}}_F\cdot\boldsymbol{\tau},$$
$$\forall(r,\boldsymbol{\tau}) \in \mathcal{P}^{0,k+1}(F)\times\boldsymbol{\mathcal{R}}^{\mathrm{c},k}(F).$$

Similar considerations apply to the element curl and vector potential. For $T \in \mathcal{T}_h$, the element curl $\mathbf{C}_T^k : \underline{\boldsymbol{X}}^k_{\mathbf{curl},T} \to \boldsymbol{\mathcal{P}}^k(T)$ is defined by: For all $\underline{\boldsymbol{v}}_T \in \underline{\boldsymbol{X}}^k_{\mathbf{curl},T}$,

$$\int_T \mathbf{C}_T^k\underline{\boldsymbol{v}}_T\cdot\boldsymbol{w} = \int_T \boldsymbol{v}_{\mathcal{R},T}\cdot\mathbf{curl}\,\boldsymbol{w} + \sum_{F\in\mathcal{F}_T}\omega_{TF}\int_F \boldsymbol{\gamma}^k_{\mathrm{t},F}\underline{\boldsymbol{v}}_F\cdot(\boldsymbol{w}\times\boldsymbol{n}_F), \qquad \forall\boldsymbol{w}\in\boldsymbol{\mathcal{P}}^k(T).$$

The vector potential $\boldsymbol{P}^k_{\mathbf{curl},T} : \underline{\boldsymbol{X}}^k_{\mathbf{curl},T} \to \boldsymbol{\mathcal{P}}^k(T)$ is given by: For all $\underline{\boldsymbol{v}}_T \in \underline{\boldsymbol{X}}^k_{\mathbf{curl},T}$,

$$\int_T \boldsymbol{P}^k_{\mathbf{curl},T}\underline{\boldsymbol{v}}_T \cdot (\mathbf{curl}\,\boldsymbol{w} + \boldsymbol{\tau}) = \int_T \mathbf{C}^k_T\underline{\boldsymbol{v}}_T \cdot \boldsymbol{w} - \sum_{F\in\mathcal{F}_T} \omega_{TF}\int_F \boldsymbol{\gamma}^k_{\mathrm{t},F}\underline{\boldsymbol{v}}_F \cdot (\boldsymbol{w}\times\boldsymbol{n}_F) + \int_T \boldsymbol{S}^k_{\mathbf{curl},T}\underline{\boldsymbol{v}}_T \cdot \boldsymbol{\tau},$$
$$\forall (\boldsymbol{w},\boldsymbol{\tau}) \in \boldsymbol{\mathcal{G}}^{\mathrm{c},k+1}(T)\times\boldsymbol{\mathcal{R}}^{\mathrm{c},k}(T).$$

2.2.4. Operators on the divergence space

The discrete divergence and potential on $\underline{\boldsymbol{X}}^k_{\mathrm{div},T}$, for $T \in \mathcal{T}_h$, are identical to those of the DDR complex since no serendipity reduction is actually possible on this space (see [1, Section 6.5]): $D^k_T : \underline{\boldsymbol{X}}^k_{\mathrm{div},T} \to \mathcal{P}^k(T)$ and $\boldsymbol{P}^k_{\mathrm{div},T} : \underline{\boldsymbol{X}}^k_{\mathrm{div},T} \to \boldsymbol{\mathcal{P}}^k(T)$ are such that, for all $\underline{\boldsymbol{w}}_T \in \underline{\boldsymbol{X}}^k_{\mathrm{div},T}$,

$$\int_T D^k_T\underline{\boldsymbol{w}}_T q = -\int_T \boldsymbol{w}_{\mathcal{G},T}\cdot\mathbf{grad}\,q + \sum_{F\in\mathcal{F}_T}\omega_{TF}\int_F w_F q, \qquad \forall q\in\mathcal{P}^k(T),$$
$$\int_T \boldsymbol{P}^k_{\mathrm{div},T}\underline{\boldsymbol{w}}_T\cdot(\mathbf{grad}\,r+\boldsymbol{\tau}) = -\int_T D^k_T\underline{\boldsymbol{w}}_T r + \sum_{F\in\mathcal{F}_T}\omega_{TF}\int_F w_F r + \int_T \boldsymbol{w}^{\mathrm{c}}_{\mathcal{G},T}\cdot\boldsymbol{\tau},$$
$$\forall (r,\boldsymbol{\tau}) \in \mathcal{P}^{0,k+1}(T)\times\boldsymbol{\mathcal{G}}^{\mathrm{c},k}(T).$$

2.2.5. Serendipity DDR complex

The serendipity DDR complex is

$$\mathbb{R} \xrightarrow{I^k_{\mathbf{grad},h}} \underline{X}^k_{\mathbf{grad},h} \xrightarrow{\underline{\boldsymbol{G}}^k_h} \underline{\boldsymbol{X}}^k_{\mathbf{curl},h} \xrightarrow{\underline{\boldsymbol{C}}^k_h} \underline{\boldsymbol{X}}^k_{\mathrm{div},h} \xrightarrow{D^k_h} \mathcal{P}^k(\mathcal{T}_h) \xrightarrow{0} \{0\},$$

where the discrete differential operators $\underline{\boldsymbol{G}}^k_h$, $\underline{\boldsymbol{C}}^k_h$ and D^k_h are obtained projecting the edge/face/element operators onto the proper spaces (dictated by the co-domains):

$$\begin{aligned}
\underline{\boldsymbol{G}}^k_h\underline{q}_h &\coloneqq \big((\boldsymbol{\pi}^{k-1}_{\mathcal{R},T}\mathbf{G}^k_T\underline{q}_T, \boldsymbol{\pi}^{\mathrm{c},\ell_T+1}_{\mathcal{R},T}\mathbf{G}^k_T\underline{q}_T)_{T\in\mathcal{T}_h}, (\boldsymbol{\pi}^{k-1}_{\mathcal{R},F}\mathbf{G}^k_F\underline{q}_F, \boldsymbol{\pi}^{\mathrm{c},\ell_F+1}_{\mathcal{R},F}\mathbf{G}^k_F\underline{q}_F)_{F\in\mathcal{F}_h}, (G^k_E\underline{q}_E)_{E\in\mathcal{E}_h}\big),\\
\underline{\boldsymbol{C}}^k_h\underline{\boldsymbol{v}}_h &\coloneqq \big((\boldsymbol{\pi}^{k-1}_{\mathcal{G},T}\mathbf{C}^k_T\underline{\boldsymbol{v}}_T, \boldsymbol{\pi}^{\mathrm{c},k}_{\mathcal{G},T}\mathbf{C}^k_T\underline{\boldsymbol{v}}_T)_{T\in\mathcal{T}_h}, (C^k_F\underline{\boldsymbol{v}}_F)_{F\in\mathcal{F}_h}\big),\\
D^k_h\underline{\boldsymbol{w}}_h &\coloneqq \big(D^k_T\underline{\boldsymbol{w}}_T\big)_{T\in\mathcal{T}_h}.
\end{aligned}$$

It was proved that this sequence is indeed a complex [11, 12], and has the same cohomology as the de Rham complex [30].

2.2.6. Discrete L^2-inner products

To design numerical schemes based on the SDDR complex, an essential ingredient, besides the discrete differential operators, are consistent L^2-inner products on the spaces of the complex. A scheme can then be designed by replacing, in the weak formulation of the PDE, the continuous differential operators and L^2-products by the discrete operators of the complex and the L^2-inner products on its spaces.

The design of these discrete L^2-inner products rely on the element potential reconstructions defined in the previous sections. Specifically, if $\underline{X}^k_{\bullet,h}$ is one of the space $\underline{X}^k_{\mathbf{grad},h}$, $\underline{\boldsymbol{X}}^k_{\mathbf{curl},h}$ or $\underline{\boldsymbol{X}}^k_{\mathrm{div},h}$ and $P^k_{\bullet,T}$ is

the associated potential in the element T, the discrete L^2-product on $\underline{X}^k_{\bullet,h}$ is defined by

$$(\underline{x}_h, \underline{y}_h)_{\bullet,h} \coloneqq \sum_{T\in\mathcal{T}_h} (\underline{x}_T, \underline{y}_T)_{\bullet,T} \quad \text{with} \quad (\underline{x}_T, \underline{y}_T)_{\bullet,T} \coloneqq \left[\int_T P^k_{\bullet,T}\underline{x}_T \cdot P^k_{\bullet,T}\underline{y}_T + \mathrm{s}_{\bullet,T}(\underline{x}_T, \underline{y}_T)\right],$$

where the dot product in the integral is replaced by a multiplication if $\bullet = \mathbf{grad}$, and the stabilisation term $\mathrm{s}_{\bullet,T}$ penalises the difference between traces of the element potential and potential reconstructions on the face/edges (where relevant). The precise definition of the stabilisation term therefore depends on the space, and the available traces:

$$\begin{aligned}
\mathrm{s}_{\mathbf{grad},T}(\underline{r}_T, \underline{q}_T) \coloneqq{}& \sum_{F\in\mathcal{F}_T} h_F \int_F \big(P^{k+1}_{\mathbf{grad},T}\underline{r}_T - \gamma^{k+1}_F \underline{r}_F\big)\big(P^{k+1}_{\mathbf{grad},T}\underline{q}_T - \gamma^{k+1}_F \underline{q}_F\big) \\
&+ \sum_{E\in\mathcal{E}_T} h_E^2 \int_E \big(P^{k+1}_{\mathbf{grad},T}\underline{r}_T - \gamma^{k+1}_E \underline{r}_E\big)\big(P^{k+1}_{\mathbf{grad},T}\underline{q}_T - \gamma^{k+1}_E \underline{q}_E\big), \quad \forall \underline{r}_T, \underline{q}_T \in \underline{X}^k_{\mathbf{grad},T}, \\
\mathrm{s}_{\mathbf{curl},T}(\underline{\boldsymbol{w}}_T, \underline{\boldsymbol{v}}_T) \coloneqq{}& \sum_{F\in\mathcal{F}_T} h_F \int_F \big((\boldsymbol{P}^k_{\mathbf{curl},T}\underline{\boldsymbol{w}}_T)_{\mathrm{t},F} - \boldsymbol{\gamma}^k_{\mathrm{t},F}\underline{\boldsymbol{w}}_F\big) \cdot \big((\boldsymbol{P}^k_{\mathbf{curl},T}\underline{\boldsymbol{v}}_T)_{\mathrm{t},F} - \boldsymbol{\gamma}^k_{\mathrm{t},F}\underline{\boldsymbol{v}}_F\big) \\
&+ \sum_{E\in\mathcal{E}_T} h_E^2 \int_E \big(\boldsymbol{P}^k_{\mathbf{curl},T}\underline{\boldsymbol{w}}_T \cdot \boldsymbol{t}_E - w_E\big)\big(\boldsymbol{P}^k_{\mathbf{curl},T}\underline{\boldsymbol{v}}_T \cdot \boldsymbol{t}_E - v_E\big), \quad \forall \underline{\boldsymbol{w}}_T, \underline{\boldsymbol{v}}_T \in \underline{\boldsymbol{X}}^k_{\mathbf{curl},T}, \\
\mathrm{s}_{\mathrm{div},T}(\underline{\boldsymbol{w}}_T, \underline{\boldsymbol{v}}_T) \coloneqq{}& \sum_{F\in\mathcal{F}_T} h_F \int_F \big(\boldsymbol{P}^k_{\mathrm{div},T}\underline{\boldsymbol{w}}_T \cdot \boldsymbol{n}_F - w_F\big)\big(\boldsymbol{P}^k_{\mathrm{div},T}\underline{\boldsymbol{v}}_T \cdot \boldsymbol{n}_F - v_F\big), \quad \forall \underline{\boldsymbol{w}}_T, \underline{\boldsymbol{v}}_T \in \underline{\boldsymbol{X}}^k_{\mathrm{div},T}. \quad (2.1)
\end{aligned}$$

An important property of the potential reconstruction on each mesh entity is their polynomial consistency: Applied to interpolates of polynomials of the correct degree ℓ ($\ell = k+1$ for the gradient space, $\ell = k$ for the curl and divergence spaces), they return the polynomial itself. This translates into the following polynomial consistency of the L^2-inner products:

$$(\underline{I}^k_{\bullet,T} f, \underline{I}^k_{\bullet,T} g)_{\bullet,T} = \int_T f \cdot g, \qquad \forall f, g \in \mathcal{P}^\ell(T).$$

2.3. Lie algebra-valued serendipity DDR complex

Since the Yang–Mills equations involve Lie algebra-valued functions, a Lie algebra-valued complex is required to discretise them. This complex is simply obtained by tensorisation of the real-valued complex, as in [2]: the spaces are made of Lie algebra-valued polynomials, and the operators of the complex act component by component on the Lie algebra.

In the following, we consider a Lie algebra $\mathfrak{g}$, that is, a finite-dimensional vector space endowed with a bilinear bracket $[\cdot,\cdot] : \mathfrak{g}\times\mathfrak{g} \to \mathfrak{g}$ and an inner product $\langle\cdot,\cdot\rangle : \mathfrak{g}\times\mathfrak{g} \to \mathbb{R}$ which satisfy the Jacobi identity

$$[a,[b,c]] + [b,[c,a]] + [c,[a,b]] = 0, \quad \forall a,b,c \in \mathfrak{g}$$

and the Ad-invariance property, which implies

$$\langle [a,b], c\rangle = \langle a, [b,c]\rangle, \quad \forall a,b,c \in \mathfrak{g}.$$

We denote the Lie algebra-valued SDDR spaces by appending an exponent $\mathfrak{g}$ after the degree k. So, for example, the gradient space in the LASDDR (Lie algebra SDDR) complex is

$$\underline{X}^{k,\mathfrak{g}}_{\mathbf{grad},h} \coloneqq (\underline{X}^{k}_{\mathbf{grad},h} \otimes \mathfrak{g}) \equiv \Big\{ \underline{q}_h = ((q_T)_{T\in\mathcal{T}_h}, (q_F)_{F\in\mathcal{F}_h}, (q_E)_{E\in\mathcal{E}_h}, (q_V)_{V\in\mathcal{V}_h}) \,:\, q_T \in \mathcal{P}^{\ell_T}(T)\otimes\mathfrak{g} \text{ for all } T\in\mathcal{T}_h,\ q_F \in \mathcal{P}^{\ell_F}(F)\otimes\mathfrak{g} \text{ for all } F\in\mathcal{F}_h,\ q_E \in \mathcal{P}^{k-1}(E)\otimes\mathfrak{g} \text{ for all } E\in\mathcal{E}_h, \text{ and } q_V \in \mathbb{R}\otimes\mathfrak{g} \text{ for all } V\in\mathcal{V}_h \Big\}.$$

We note that, selecting a basis $(e_I)_I$ of $\mathfrak{g}$, for any $\mathsf{P}\in\mathcal{M}_h$ we have

$$\mathcal{P}^\ell(\mathsf{P})\otimes\mathfrak{g} \equiv \mathcal{P}^\ell(\mathsf{P};\mathfrak{g}) \coloneqq \{\phi^I e_I \,:\, \phi^I \in \mathcal{P}^\ell(\mathsf{P})\}.$$

Here and in the following we use the implicit summation convention so, for example, $\phi^I e_I = \sum_I \phi^I e_I$. For a general space X, an element $v\in X\otimes\mathfrak{g}$ can be uniquely decomposed as $v = v^I\otimes e_I$. Any linear operator $L : X\to Y$ acting between two SDDR spaces X, Y (or an SDDR space and a polynomial space) – such as a discrete differential operator, a potential reconstruction, etc. – then gives rise to the corresponding LASDDR operator $L^{\mathfrak{g}} : X\otimes\mathfrak{g}\to Y\otimes\mathfrak{g}$ defined as $L^{\mathfrak{g}}(v) = (L(v^I))\otimes e_I \in Y\otimes\mathfrak{g}$; this definition is independent of the choice of the basis in $\mathfrak{g}$. With these notations, the LASDDR complex is

$$\mathbb{R}\otimes\mathfrak{g} \xrightarrow{\underline{I}^{k,\mathfrak{g}}_{\mathbf{grad},h}} \underline{X}^{k,\mathfrak{g}}_{\mathbf{grad},h} \xrightarrow{\underline{\boldsymbol{G}}^{k,\mathfrak{g}}_{h}} \underline{\boldsymbol{X}}^{k,\mathfrak{g}}_{\mathbf{curl},h} \xrightarrow{\underline{\boldsymbol{C}}^{k,\mathfrak{g}}_{h}} \underline{\boldsymbol{X}}^{k,\mathfrak{g}}_{\mathrm{div},h} \xrightarrow{D^{k,\mathfrak{g}}_{h}} \mathcal{P}^k(\mathcal{T}_h)\otimes\mathfrak{g} \xrightarrow{0} \{0\}.$$

In each space an inner product is obtained by tensorising the inner product of the corresponding SDDR space and of the Lie algebra. So, if $\bullet\in\{\mathbf{grad}, \mathbf{curl}, \mathrm{div}\}$,

$$(\underline{x}_h, \underline{y}_h)_{\bullet,\mathfrak{g},h} = (\underline{x}^I_h, \underline{y}^J_h)_{\bullet,h}\langle e_I, e_J\rangle, \qquad \forall \underline{x}_h = \underline{x}^I_h\otimes e_I \in \underline{X}^{k,\mathfrak{g}}_{\bullet,h}, \quad \forall \underline{y}_h = \underline{y}^J_h\otimes e_J \in \underline{X}^{k,\mathfrak{g}}_{\bullet,h}.$$

Practical implementations of the LASDDR complex and related schemes can be easily done, in principle, by tensorising the operators and inner products of an SDDR implementation. Early tensorisation can however lead to unduly expensive calculations, especially when nonlinear terms are involved. We discuss in Section 4 the main considerations that must be taken into account to limit the assembly cost in implementations of LASDDR-based schemes.

3. Two DDR-based schemes for the Yang–Mills equations

We propose two schemes for the Yang–Mills equations, which only differ in the handling of the nonlinear terms appearing in the equations. The first was introduced at the lowest order in [2], in which a discrete 'bracket' was constructed to approximate the value in the $\underline{\boldsymbol{X}}^{k,\mathfrak{g}}_{\mathrm{div},h}$ space. This term is used in the discrete L^2-products, and the exact preservation of a discrete constraint, as well as energy estimates, are proven. Numerical tests in [2] however only considered the lowest-order $k=0$ of the method.

The second method we present here is new, and leverages instead the continuous L^2-product and nonlinear bracket, as well as the elemental potential reconstructions, to achieve the same goal.

3.1. Weak constrained form of the equations

Deriving from the bracket on the Lie algebra, the following two bilinear maps are defined:

$$[\cdot,\cdot] : (\mathfrak{X}(U)\otimes\mathfrak{g})\times(C^\infty(U)\otimes\mathfrak{g}) \to \mathfrak{X}(U)\otimes\mathfrak{g}, \qquad [\boldsymbol{v}, q] \mapsto \boldsymbol{v}^I q^J \otimes [e_I, e_J], \tag{3.1}$$

$$\star[\cdot,\cdot] : (\mathfrak{X}(U)\otimes\mathfrak{g})\times(\mathfrak{X}(U)\otimes\mathfrak{g}) \to \mathfrak{X}(U)\otimes\mathfrak{g}, \qquad \star[\boldsymbol{v}, \boldsymbol{w}] \mapsto (\boldsymbol{v}^I \times \boldsymbol{w}^J) \otimes [e_I, e_J]. \tag{3.2}$$

We use in the discretisation a weak *constrained* formulation of the Yang–Mills equations, appearing previously in [20]: Find $(\boldsymbol{A}, \boldsymbol{E}, \lambda) : [0,T] \to (\boldsymbol{H}(\mathbf{curl}; U)\otimes\mathfrak{g})^2 \times (H^1(U)\otimes\mathfrak{g})$ such that

$$\partial_t \boldsymbol{A} = -\boldsymbol{E}, \tag{3.3a}$$

$$\int_U \langle \partial_t \boldsymbol{E}, \boldsymbol{v}\rangle + \int_U \langle \mathbf{grad}\,\lambda + [\boldsymbol{A},\lambda], \boldsymbol{v}\rangle = \int_U \langle \mathbf{curl}\,\boldsymbol{A}, \mathbf{curl}\,\boldsymbol{v}\rangle + \int_U \langle \mathbf{curl}\,\boldsymbol{A}, \star[\boldsymbol{A},\boldsymbol{v}]\rangle + \int_U \left\langle \frac{1}{2}\star[\boldsymbol{A},\boldsymbol{A}], \mathbf{curl}\,\boldsymbol{v} + \star[\boldsymbol{A},\boldsymbol{v}]\right\rangle, \qquad \forall \boldsymbol{v} \in \boldsymbol{H}(\mathbf{curl}; U)\otimes\mathfrak{g}, \tag{3.3b}$$

$$\int_U \langle \partial_t \boldsymbol{E}, \mathbf{grad}\, q + [\boldsymbol{A}, q]\rangle = 0, \qquad \forall q \in H^1(U)\otimes\mathfrak{g}. \tag{3.3c}$$

Note that the right-hand side of (3.3b) is equal to $\int_U \langle \boldsymbol{B}, \mathbf{curl}\,\boldsymbol{v} + \star[\boldsymbol{A},\boldsymbol{v}]\rangle$, where the magnetic field is defined as $\boldsymbol{B} \coloneqq \mathbf{curl}\,\boldsymbol{A} + \frac{1}{2}\star[\boldsymbol{A},\boldsymbol{A}]$. We have developed this expression as it will drive different choices of discretisations. Solutions to these equations preserve the quantity

$$\int_U \langle \boldsymbol{E}, \mathbf{grad}\, q + [\boldsymbol{A}, q]\rangle, \qquad \forall q \in H^1(U)\otimes\mathfrak{g}. \tag{3.4}$$

The use of this particular constrained form facilitates the preservation of a discrete counterpart in the numerical scheme; discussions on the derivation and implications of these continuous equations can be found in [2, 20].

3.2. Schemes

We consider a time discretisation $0 = t^0 < t^1 < \ldots < t^N = T$ of $[0,T]$ and denote the step in time between n and $n+1$ as $\delta t^{n+\frac{1}{2}} \coloneqq t^{n+1} - t^n$. Then define for a family $v = (v^n)_n$, $\delta_t^{n+1} v = \frac{v^{n+1}-v^n}{\delta t^{n+\frac{1}{2}}}$.

Starting from initial conditions $(\underline{\boldsymbol{A}}_h^0, \underline{\boldsymbol{E}}_h^0) \in (\underline{\boldsymbol{X}}^{k,\mathfrak{g}}_{\mathbf{curl},h})^2$, the constrained scheme based on (3.3) is: Find families $(\underline{\boldsymbol{A}}_h^n)_n$, $(\underline{\boldsymbol{E}}_h^n)_n$, $(\underline{\lambda}_h^n)_n$ such that for all n, $(\underline{\boldsymbol{A}}_h^n, \underline{\boldsymbol{E}}_h^n, \underline{\lambda}_h^n) \in (\underline{\boldsymbol{X}}^{k,\mathfrak{g}}_{\mathbf{curl},h})^2 \times (\underline{X}^{k,\mathfrak{g}}_{\mathbf{grad},h})$ and

$$\delta_t^{n+1} \underline{\boldsymbol{A}}_h = -\underline{\boldsymbol{E}}_h^{n+1}, \tag{3.5a}$$

$$(\delta_t^{n+1}\underline{\boldsymbol{E}}_h, \underline{\boldsymbol{v}}_h)_{\mathbf{curl},\mathfrak{g},h} + (\underline{\boldsymbol{G}}_h^{k,\mathfrak{g}}\underline{\lambda}_h^{n+1}, \underline{\boldsymbol{v}}_h)_{\mathbf{curl},\mathfrak{g},h} + \int_U \langle [\boldsymbol{P}^{k,\mathfrak{g}}_{\mathbf{curl},h}\underline{\boldsymbol{A}}_h^{n+1}, P^{k+1,\mathfrak{g}}_{\mathbf{grad},h}\underline{\lambda}_h^{n+1}], \boldsymbol{P}^{k,\mathfrak{g}}_{\mathbf{curl},h}\underline{\boldsymbol{v}}_h\rangle = (\underline{\boldsymbol{C}}_h^{k,\mathfrak{g}}\underline{\boldsymbol{A}}_h^{n+1}, \underline{\boldsymbol{C}}_h^{k,\mathfrak{g}}\underline{\boldsymbol{v}}_h)_{\mathrm{div},\mathfrak{g},h} + \mathfrak{N}(\underline{\boldsymbol{A}}_h^n, \underline{\boldsymbol{A}}_h^{n+1}; \underline{\boldsymbol{v}}_h), \qquad \forall \underline{\boldsymbol{v}}_h \in \underline{\boldsymbol{X}}^{k,\mathfrak{g}}_{\mathbf{curl},h}, \tag{3.5b}$$

$$(\delta_t^{n+1}\underline{\boldsymbol{E}}_h, \underline{\boldsymbol{G}}_h^{k,\mathfrak{g}}\underline{q}_h)_{\mathbf{curl},\mathfrak{g},h} + \int_U \langle \boldsymbol{P}^{k,\mathfrak{g}}_{\mathbf{curl},h}(\delta_t^{n+1}\underline{\boldsymbol{E}}_h), [\boldsymbol{P}^{k,\mathfrak{g}}_{\mathbf{curl},h}\underline{\boldsymbol{A}}_h^n, P^{k+1,\mathfrak{g}}_{\mathbf{grad},h}\underline{q}_h]\rangle = 0, \qquad \forall \underline{q}_h \in \underline{X}^{k,\mathfrak{g}}_{\mathbf{grad},h}. \tag{3.5c}$$

We refer the reader to [2, Section 4] for a discussion on the choice of the initial conditions $(\underline{\boldsymbol{A}}_h^0, \underline{\boldsymbol{E}}_h^0)$, and also for alternative choices to the fully implicit time stepping selected here.

In (3.5b), $\mathfrak{N} \in \{\mathfrak{N}_1, \mathfrak{N}_2\}$ is one of the following two discretisations of the nonlinear terms in the right-hand side of (3.3b):

$$\begin{aligned}\mathfrak{N}_1(\underline{\boldsymbol{A}}_h^n, \underline{\boldsymbol{A}}_h^{n+1}; \underline{\boldsymbol{v}}_h) &\coloneqq (\underline{\boldsymbol{C}}_h^{k,\mathfrak{g}} \underline{\boldsymbol{A}}_h^{n+1}, \star[\underline{\boldsymbol{A}}_h^{n+\frac12}, \underline{\boldsymbol{v}}_h]^{\mathrm{div},k,h})_{\mathrm{div},\mathfrak{g},h} \\ &\quad + \left(\frac12 \star[\underline{\boldsymbol{A}}_h^{n+1}, \underline{\boldsymbol{A}}_h^{n+1}]^{\mathrm{div},k,h}, \underline{\boldsymbol{C}}_h^{k,\mathfrak{g}} \underline{\boldsymbol{v}}_h + \star[\underline{\boldsymbol{A}}_h^{n+\frac12}, \underline{\boldsymbol{v}}_h]^{\mathrm{div},k,h}\right)_{\mathrm{div},\mathfrak{g},h}, \qquad (3.6)\\ \mathfrak{N}_2(\underline{\boldsymbol{A}}_h^n, \underline{\boldsymbol{A}}_h^{n+1}; \underline{\boldsymbol{v}}_h) &\coloneqq \int_U \langle \mathbf{C}_h^{k,\mathfrak{g}} \underline{\boldsymbol{A}}_h^{n+1}, \star[\boldsymbol{P}_{\mathbf{curl},h}^{k,\mathfrak{g}} \underline{\boldsymbol{A}}_h^{n+\frac12}, \boldsymbol{P}_{\mathbf{curl},h}^{k,\mathfrak{g}} \underline{\boldsymbol{v}}_h]\rangle \\ &\quad + \int_U \left\langle \frac12 \star[\boldsymbol{P}_{\mathbf{curl},h}^{k,\mathfrak{g}} \underline{\boldsymbol{A}}_h^{n+1}, \boldsymbol{P}_{\mathbf{curl},h}^{k,\mathfrak{g}} \underline{\boldsymbol{A}}_h^{n+1}], \mathbf{C}_h^{k,\mathfrak{g}} \underline{\boldsymbol{v}}_h + \star[\boldsymbol{P}_{\mathbf{curl},h}^{k,\mathfrak{g}} \underline{\boldsymbol{A}}_h^{n+\frac12}, \boldsymbol{P}_{\mathbf{curl},h}^{k,\mathfrak{g}} \underline{\boldsymbol{v}}_h]\right\rangle. \qquad (3.7)\end{aligned}$$

Above, we have set $\underline{\boldsymbol{A}}_h^{n+\frac12} = \frac12(\underline{\boldsymbol{A}}_h^n + \underline{\boldsymbol{A}}_h^{n+1})$. Moreover, in $\mathfrak{N}_1$, we have made use of the discrete version $\star[\cdot,\cdot]^{\mathrm{div},k,h} : (\underline{\boldsymbol{X}}_{\mathbf{curl},h}^{k,\mathfrak{g}})^2 \to \underline{\boldsymbol{X}}_{\mathrm{div},h}^{k,\mathfrak{g}}$ of the map (3.2), defined for all $\underline{\boldsymbol{v}}_h, \underline{\boldsymbol{w}}_h \in \underline{\boldsymbol{X}}_{\mathbf{curl},h}^{k,\mathfrak{g}}$ through its components by:

$$\left(\star[\underline{\boldsymbol{v}}_h, \underline{\boldsymbol{w}}_h]^{\mathrm{div},k,h}\right)_F = \pi_{\mathcal{P},F}^k \left(\star[\boldsymbol{\gamma}_{\mathrm{t},F}^{k,\mathfrak{g}} \underline{\boldsymbol{v}}_F, \boldsymbol{\gamma}_{\mathrm{t},F}^{k,\mathfrak{g}} \underline{\boldsymbol{w}}_F] \cdot \boldsymbol{n}_F\right) \in \mathcal{P}^k(F) \otimes \mathfrak{g}, \quad \forall F \in \mathcal{F}_h, \qquad (3.8a)$$

$$\left(\star[\underline{\boldsymbol{v}}_h, \underline{\boldsymbol{w}}_h]^{\mathrm{div},k,h}\right)_{\mathcal{G},T} = \boldsymbol{\pi}_{\mathcal{G},T}^{k-1} \left(\star[\boldsymbol{P}_{\mathbf{curl},T}^{k,\mathfrak{g}} \underline{\boldsymbol{v}}_T, \boldsymbol{P}_{\mathbf{curl},T}^{k,\mathfrak{g}} \underline{\boldsymbol{w}}_T]\right) \in \boldsymbol{\mathcal{G}}^{k-1}(T) \otimes \mathfrak{g}, \quad \forall T \in \mathcal{T}_h, \qquad (3.8b)$$

$$\left(\star[\underline{\boldsymbol{v}}_h, \underline{\boldsymbol{w}}_h]^{\mathrm{div},k,h}\right)_{\mathcal{G},T}^{\mathrm{c}} = \boldsymbol{\pi}_{\mathcal{G},T}^{\mathrm{c},k} \left(\star[\boldsymbol{P}_{\mathbf{curl},T}^{k,\mathfrak{g}} \underline{\boldsymbol{v}}_T, \boldsymbol{P}_{\mathbf{curl},T}^{k,\mathfrak{g}} \underline{\boldsymbol{w}}_T]\right) \in \boldsymbol{\mathcal{G}}^{\mathrm{c},k}(T) \otimes \mathfrak{g}, \quad \forall T \in \mathcal{T}_h. \qquad (3.8c)$$

In $\mathfrak{N}_2$ we have used the global piecewise polynomial curl $\mathbf{C}_h^{k,\mathfrak{g}}$ defined by patching the element curls: $(\mathbf{C}_h^{k,\mathfrak{g}} \underline{\boldsymbol{v}}_h)_{|T} = \mathbf{C}_T^{k,\mathfrak{g}} \underline{\boldsymbol{v}}_T$ for all $T \in \mathcal{T}_h$ and $\underline{\boldsymbol{v}}_h \in \underline{\boldsymbol{X}}_{\mathbf{curl},h}^{k,\mathfrak{g}}$.

Remark 2 (Motivation for the discretisation of the nonlinear terms). The nonlinear terms in the right-hand side of (3.3b) are

$$\int_U \langle \mathbf{curl}\, \boldsymbol{A}, \star[\boldsymbol{A}, \boldsymbol{v}]\rangle + \int_U \left\langle \frac12 \star[\boldsymbol{A}, \boldsymbol{A}], \mathbf{curl}\, \boldsymbol{v} + \star[\boldsymbol{A}, \boldsymbol{v}]\right\rangle. \qquad (3.9)$$

When discretising these terms, the continuous fields $\boldsymbol{A}, \boldsymbol{v} \in \boldsymbol{H}(\mathbf{curl}; U) \otimes \mathfrak{g}$ are replaced by fully discrete objects $\underline{\boldsymbol{A}}_h, \underline{\boldsymbol{v}}_h \in \underline{\boldsymbol{X}}_{\mathbf{curl},h}^{k,\mathfrak{g}}$, and we have to give meaning to the terms in (3.9) after this substitution – which is not straightforward since $\underline{\boldsymbol{X}}_{\mathbf{curl},h}^{k,\mathfrak{g}}$ is not a subspace of $\boldsymbol{H}(\mathbf{curl}; U) \otimes \mathfrak{g}$.

Applying the standard DDR procedure on (3.9), we build these terms by replacing the inner product $\int_U \langle\cdot,\cdot\rangle$ and differential **curl** by the corresponding discrete notions found in the LASDDR complex. The only missing element is a discrete version of the bracket $\star[\cdot,\cdot]$ on $\underline{\boldsymbol{X}}_{\mathbf{curl},h}^{k,\mathfrak{g}} \times \underline{\boldsymbol{X}}_{\mathbf{curl},h}^{k,\mathfrak{g}}$ which produces consistent discrete approximations in $\underline{\boldsymbol{X}}_{\mathrm{div},h}^{k,\mathfrak{g}}$. This is what (3.8) provides, and this approach leads to $\mathfrak{N}_1$.

Another approach to discretising (3.9) is in a sense more straightforward (but only works because none of these terms, in the weak formulation, comes from integrating-by-parts the strong form of the model): Since we can reconstruct piecewise polynomial reconstructions and curls from elements in $\underline{\boldsymbol{X}}_{\mathbf{curl},h}^{k,\mathfrak{g}}$, we can decide to simply substitute all the terms $\boldsymbol{A}, \boldsymbol{v}$ by these polynomial reconstruction based on $\underline{\boldsymbol{A}}_h, \underline{\boldsymbol{v}}_h$ and keep the other elements (integrals, brackets) exactly the same. This idea leads to $\mathfrak{N}_2$.

Remark 3 (Discretisation of the linear terms). The same way we used, in $\mathfrak{N}_2$, the piecewise polynomial potentials and element curl, we could consider replacing, in (3.5b), the term $(\underline{\boldsymbol{C}}_h^{k,\mathfrak{g}}\underline{\boldsymbol{A}}_h^{n+1}, \underline{\boldsymbol{C}}_h^{k,\mathfrak{g}}\underline{\boldsymbol{v}}_h)_{\mathrm{div},\mathfrak{g},h}$ with $\int_U \langle \mathbf{C}_h^{k,\mathfrak{g}}\underline{\boldsymbol{A}}_h^{n+1}, \mathbf{C}_h^{k,\mathfrak{g}}\underline{\boldsymbol{v}}_h\rangle$. This would however not lead to a suitable scheme, for the following reason.

Consider the pure Maxwell model, discretised using a linear unconstrained scheme (that is, (3.5a)–(3.5b) without the terms involving $\underline{\lambda}_h$, without the nonlinear terms and with $\mathfrak{g} = \mathbb{R}$ with the trivial Lie bracket):

$$\delta_t^{n+1}\underline{\boldsymbol{A}}_h = -\underline{\boldsymbol{E}}_h^{n+1}, \tag{3.10a}$$

$$(\delta_t^{n+1}\underline{\boldsymbol{E}}_h, \underline{\boldsymbol{v}}_h)_{\mathbf{curl},\mathfrak{g},h} = (\underline{\boldsymbol{C}}_h^{k,\mathfrak{g}}\underline{\boldsymbol{A}}_h^{n+1}, \underline{\boldsymbol{C}}_h^{k,\mathfrak{g}}\underline{\boldsymbol{v}}_h)_{\mathrm{div},\mathfrak{g},h}. \tag{3.10b}$$

Using the results in [12, Section 6] it can easily be shown that, for a smooth enough potential $\boldsymbol{A}$, solution of the continuous model, the consistency error (as defined in [31]) of the scheme satisfies

$$\mathcal{E}_h(\boldsymbol{A}; \underline{\boldsymbol{v}}_h) \le C_{\boldsymbol{A}}(\delta t + h^{k+1})\left(\|\underline{\boldsymbol{v}}_h\|_{\mathbf{curl},\mathfrak{g},h} + \|\underline{\boldsymbol{C}}_h^k\underline{\boldsymbol{v}}_h\|_{\mathrm{div},\mathfrak{g},h}\right), \tag{3.11}$$

where $\|\cdot\|_{\mathbf{curl},\mathfrak{g},h}$ and $\|\cdot\|_{\mathrm{div},\mathfrak{g},h}$ denote the norms respectively associated with the inner products $(\cdot,\cdot)_{\mathbf{curl},\mathfrak{g},h}$ and $(\cdot,\cdot)_{\mathrm{div},\mathfrak{g},h}$. The scheme (3.10) is stable for the norm $\|\cdot\|_{\mathbf{curl},\mathfrak{g},h} + \|\underline{\boldsymbol{C}}_h^k\cdot\|_{\mathrm{div},\mathfrak{g},h}$, so (3.11) and the 3rd Strang Lemma [31] provide an $O(\delta t + h^{k+1})$ error estimate.

However, replacing $(\underline{\boldsymbol{C}}_h^{k,\mathfrak{g}}\underline{\boldsymbol{A}}_h^{n+1}, \underline{\boldsymbol{C}}_h^{k,\mathfrak{g}}\underline{\boldsymbol{v}}_h)_{\mathrm{div},\mathfrak{g},h}$ with $\int_U \langle \mathbf{C}_h^{k,\mathfrak{g}}\underline{\boldsymbol{A}}_h^{n+1}, \mathbf{C}_h^{k,\mathfrak{g}}\underline{\boldsymbol{v}}_h\rangle$ in (3.10b) results in a scheme that is stable for the weaker norm $\|\cdot\|_{\mathbf{curl},\mathfrak{g},h} + \|\mathbf{C}_h^{k,\mathfrak{g}}\cdot\|_{L^2(U)}$ (which does not control, in particular, the face curls). On the other hand, the consistency estimate remains (3.11), in the stronger norm. As a consequence, this estimate and the weaker stability cannot be combined together to obtain error estimates on the scheme. As a matter of fact, numerical tests (not reported in this paper) show that, on some mesh families, this alternative scheme does not converge as the mesh size and time step are refined.

3.3. Discrete energy and constraint preservation

We define the discrete conserved quantity through the constraint functional $\mathfrak{C}^n : \underline{X}_{\mathbf{grad},h}^{k,\mathfrak{g}} \to \mathbb{R}$:

$$\mathfrak{C}^n(\underline{q}_h) \coloneqq (\underline{\boldsymbol{E}}_h^n, \underline{\boldsymbol{G}}_h^{k,\mathfrak{g}}\underline{q}_h)_{\mathbf{curl},\mathfrak{g},h} + \int_U \langle \boldsymbol{P}_{\mathbf{curl},h}^{k,\mathfrak{g}}\underline{\boldsymbol{E}}_h^n, [\boldsymbol{P}_{\mathbf{curl},h}^{k,\mathfrak{g}}\underline{\boldsymbol{A}}_h^n, P_{\mathbf{grad},h}^{k+1,\mathfrak{g}}\underline{q}_h]\rangle, \quad \forall \underline{q}_h \in \underline{X}_{\mathbf{grad},h}^{k,\mathfrak{g}}. \tag{3.12}$$

Proposition 4 (Constraint preservation)**.** *For any choice of* $\mathfrak{N}$*, if* $(\underline{\boldsymbol{A}}_h^n, \underline{\boldsymbol{E}}_h^n, \underline{\lambda}_h^n)$ *solve* (3.5) *then, for all* $\underline{q}_h \in \underline{X}_{\mathbf{grad},h}^{k,\mathfrak{g}}$*, the quantity* $\mathfrak{C}^n(\underline{q}_h)$ *is independent of* n*.*

Proof. We note that the proof for the preservation of constraint in [2, Proposition 7] is in fact independent of the discretisation of the second equation (3.5b), and also applies in the case for general k (see [2, Section 6.2]). □

To state the energy dissipation property, we introduce the discrete magnetic fields based on $\boldsymbol{B}$. Their nature depends on the chosen discretisation of the nonlinear terms in (3.3b). If $\mathfrak{N} = \mathfrak{N}_1$, exploiting the discrete bracket $\star[\cdot,\cdot]^{\mathrm{div},k,h}$ we can define the discrete magnetic field as an element of $\underline{\boldsymbol{X}}_{\mathbf{curl},h}^{k,\mathfrak{g}}$:

$$\underline{\boldsymbol{B}}_{h,1}^n \coloneqq \underline{\boldsymbol{C}}_h^{k,\mathfrak{g}}\underline{\boldsymbol{A}}_h^n + \frac{1}{2}\star[\underline{\boldsymbol{A}}_h^n, \underline{\boldsymbol{A}}_h^n]^{\mathrm{div},k,h} \in \underline{\boldsymbol{X}}_{\mathbf{curl},h}^{k,\mathfrak{g}}.$$

If $\mathfrak{R} = \mathfrak{R}_2$, the nonlinear terms being discretised as piecewise polynomial functions, the discrete magnetic field has the same nature:

$$\boldsymbol{B}^n_{h,2} \coloneqq \mathbf{C}^{k,\mathfrak{g}}_h \underline{\boldsymbol{A}}^n_h + \frac{1}{2}\star[\boldsymbol{P}^{k,\mathfrak{g}}_{\mathbf{curl},h}\underline{\boldsymbol{A}}^n_h, \boldsymbol{P}^{k,\mathfrak{g}}_{\mathbf{curl},h}\underline{\boldsymbol{A}}^n_h] \in \mathcal{P}^k(\mathcal{T}_h)\otimes\mathfrak{g}. \tag{3.13}$$

Proposition 5 (Energy dissipation)**.** *If the initial conditions* $(\underline{\boldsymbol{A}}^0_h, \underline{\boldsymbol{E}}^0_h)$ *are such that* $\mathfrak{C}^0 \equiv 0$*, then we have the decay of energy in the sense that, for all n,*

$$\frac{1}{2}\|\underline{\boldsymbol{E}}^{n+1}_h\|^2_{\mathbf{curl},\mathfrak{g},h} + \mathfrak{B}^{n+1} \le \frac{1}{2}\|\underline{\boldsymbol{E}}^n_h\|^2_{\mathbf{curl},\mathfrak{g},h} + \mathfrak{B}^n, \tag{3.14}$$

where

$$\mathfrak{B}^n \coloneqq \begin{cases} \dfrac{1}{2}\|\underline{\boldsymbol{B}}^n_{h,1}\|^2_{\mathrm{div},\mathfrak{g},h} & \text{if} \quad \mathfrak{R} = \mathfrak{R}_1, \\ \dfrac{1}{2}\|\boldsymbol{B}^n_{h,2}\|^2_{L^2(U)\otimes\mathfrak{g}} + \dfrac{1}{2}\mathrm{s}^{\mathfrak{g}}_{\mathrm{div},h}(\underline{\boldsymbol{C}}^{k,\mathfrak{g}}_h\underline{\boldsymbol{A}}^n_h, \underline{\boldsymbol{C}}^{k,\mathfrak{g}}_h\underline{\boldsymbol{A}}^n_h) & \text{if} \quad \mathfrak{R} = \mathfrak{R}_2, \end{cases}$$

where $\mathrm{s}^{\mathfrak{g}}_{\mathrm{div},h}$ *is the stabilisation form involved in the definition of the inner product* $(\cdot,\cdot)_{\mathrm{div},\mathfrak{g},h}$ *(that is, the tensorisation of* $\mathrm{s}_{\mathrm{div},h}$ *defined by* (2.1)*).*

Proof. The proof for $\mathfrak{R} = \mathfrak{R}_1$ is identical to the one for $k = 0$ done in [2, Proposition 8] (see also [2, Section 6.2]).

Let us consider the case $\mathfrak{R} = \mathfrak{R}_2$. Choosing $\underline{\boldsymbol{v}}_h = \underline{\boldsymbol{E}}^{n+1}_h$ in (3.5b) and multiplying by $\delta t^{n+\frac{1}{2}}$, the first term in the LHS is

$$(\underline{\boldsymbol{E}}^{n+1}_h - \underline{\boldsymbol{E}}^n_h, \underline{\boldsymbol{E}}^{n+1}_h)_{\mathbf{curl},\mathfrak{g},h} = \frac{1}{2}\|\underline{\boldsymbol{E}}^{n+1}_h\|^2_{\mathbf{curl},\mathfrak{g},h} - \frac{1}{2}\|\underline{\boldsymbol{E}}^n_h\|^2_{\mathbf{curl},\mathfrak{g},h} + \frac{1}{2}\|\underline{\boldsymbol{E}}^{n+1}_h - \underline{\boldsymbol{E}}^n_h\|^2_{\mathbf{curl},\mathfrak{g},h},$$

while the remaining two form the constraint $\mathfrak{C}^{n+1}(\underline{\lambda}^{n+1}_h)$, that vanishes by Proposition 4 and the assumption that $\mathfrak{C}^0 \equiv 0$.

On the RHS, we use (3.5a) to substitute instead $\underline{\boldsymbol{v}}_h = -\delta^{n+1}_t\underline{\boldsymbol{A}}_h$, noting the cancellation of $\delta t^{n+\frac{1}{2}}$ after multiplying. Expanding the discrete L^2-product by its definition and invoking [12, Proposition 7] (which can easily be extended to the SDDR complex using [1, Eq (2.2)]) to write $\boldsymbol{P}^{k,\mathfrak{g}}_{\mathrm{div},h}\underline{\boldsymbol{C}}^{k,\mathfrak{g}}_h = \mathbf{C}^{k,\mathfrak{g}}_h$, the first term is

$$\begin{aligned}(\underline{\boldsymbol{C}}^{k,\mathfrak{g}}_h\underline{\boldsymbol{A}}^{n+1}_h, \underline{\boldsymbol{C}}^{k,\mathfrak{g}}_h(\underline{\boldsymbol{A}}^n_h - \underline{\boldsymbol{A}}^{n+1}_h))_{\mathrm{div},\mathfrak{g},h} \\ = \int_U \langle \mathbf{C}^{k,\mathfrak{g}}_h\underline{\boldsymbol{A}}^{n+1}_h, \mathbf{C}^{k,\mathfrak{g}}_h(\underline{\boldsymbol{A}}^n_h - \underline{\boldsymbol{A}}^{n+1}_h)\rangle + \mathrm{s}^{\mathfrak{g}}_{\mathrm{div},h}(\underline{\boldsymbol{C}}^{k,\mathfrak{g}}_h\underline{\boldsymbol{A}}^n_h, \underline{\boldsymbol{C}}^{k,\mathfrak{g}}_h(\underline{\boldsymbol{A}}^n_h - \underline{\boldsymbol{A}}^{n+1}_h)).\end{aligned}$$

Combining with the integrals in $\mathfrak{R}_2$ (see (3.7)), expanding $\underline{\boldsymbol{A}}^{n+\frac{1}{2}}_h = \frac{1}{2}(\underline{\boldsymbol{A}}^n_h + \underline{\boldsymbol{A}}^{n+1}_h)$, recalling the definition (3.13) of $\boldsymbol{B}_{h,2}$, then using the symmetry and bilinearity of the bracket (3.2), the RHS becomes

$$\begin{aligned}&(\underline{\boldsymbol{C}}^{k,\mathfrak{g}}_h\underline{\boldsymbol{A}}^{n+1}_h, \underline{\boldsymbol{C}}^{k,\mathfrak{g}}_h(\underline{\boldsymbol{A}}^n_h - \underline{\boldsymbol{A}}^{n+1}_h))_{\mathrm{div},\mathfrak{g},h} + \mathfrak{R}_2(\underline{\boldsymbol{A}}^n_h, \underline{\boldsymbol{A}}^{n+1}_h; \underline{\boldsymbol{A}}^n_h - \underline{\boldsymbol{A}}^{n+1}_h) \\ &\quad = \int_U \left\langle \boldsymbol{B}^{n+1}_h, \mathbf{C}^{k,\mathfrak{g}}_h(\underline{\boldsymbol{A}}^n_h - \underline{\boldsymbol{A}}^{n+1}_h) + \star[\boldsymbol{P}^{k,\mathfrak{g}}_{\mathbf{curl},h}\frac{1}{2}(\underline{\boldsymbol{A}}^n_h + \underline{\boldsymbol{A}}^{n+1}_h), \boldsymbol{P}^{k,\mathfrak{g}}_{\mathbf{curl},h}(\underline{\boldsymbol{A}}^n_h - \underline{\boldsymbol{A}}^{n+1}_h)]\right\rangle\end{aligned}$$

$$
\begin{aligned}
&+ \mathrm{s}^{\mathfrak{g}}_{\mathrm{div},h}(\underline{\boldsymbol{C}}^{k,\mathfrak{g}}_h \underline{\boldsymbol{A}}^{n+1}_h, \underline{\boldsymbol{C}}^{k,\mathfrak{g}}_h(\underline{\boldsymbol{A}}^{n}_h - \underline{\boldsymbol{A}}^{n+1}_h)) \\
&= \int_U \left\langle \boldsymbol{B}^{n+1}_h, \mathsf{C}^{k,\mathfrak{g}}_h \underline{\boldsymbol{A}}^{n}_h + \frac{1}{2}\star[\boldsymbol{P}^{k,\mathfrak{g}}_{\mathbf{curl},h}\underline{\boldsymbol{A}}^{n}_h, \boldsymbol{P}^{k,\mathfrak{g}}_{\mathbf{curl},h}\underline{\boldsymbol{A}}^{n}_h] - \mathsf{C}^{k,\mathfrak{g}}_h \underline{\boldsymbol{A}}^{n+1}_h - \frac{1}{2}\star[\boldsymbol{P}^{k,\mathfrak{g}}_{\mathbf{curl},h}\underline{\boldsymbol{A}}^{n+1}_h, \boldsymbol{P}^{k,\mathfrak{g}}_{\mathbf{curl},h}\underline{\boldsymbol{A}}^{n+1}_h] \right\rangle \\
&\quad + \mathrm{s}^{\mathfrak{g}}_{\mathrm{div},h}(\underline{\boldsymbol{C}}^{k,\mathfrak{g}}_h \underline{\boldsymbol{A}}^{n+1}_h, \underline{\boldsymbol{C}}^{k,\mathfrak{g}}_h(\underline{\boldsymbol{A}}^{n}_h - \underline{\boldsymbol{A}}^{n+1}_h)) \\
&= \int_U \left\langle \boldsymbol{B}^{n+1}_h, \boldsymbol{B}^{n}_h - \boldsymbol{B}^{n+1}_h \right\rangle + \mathrm{s}^{\mathfrak{g}}_{\mathrm{div},h}(\underline{\boldsymbol{C}}^{k,\mathfrak{g}}_h \underline{\boldsymbol{A}}^{n+1}_h, \underline{\boldsymbol{C}}^{k,\mathfrak{g}}_h(\underline{\boldsymbol{A}}^{n}_h - \underline{\boldsymbol{A}}^{n+1}_h)) \\
&= \frac{1}{2}\Big(\|\boldsymbol{B}^{n}_h\|^2_{L^2(U)\otimes\mathfrak{g}} - \|\boldsymbol{B}^{n+1}_h\|^2_{L^2(U)\otimes\mathfrak{g}} - \|\boldsymbol{B}^{n+1}_h - \boldsymbol{B}^{n}_h\|^2_{L^2(U)\otimes\mathfrak{g}} + \mathrm{s}^{\mathfrak{g}}_{\mathrm{div},h}(\underline{\boldsymbol{C}}^{k,\mathfrak{g}}_h \underline{\boldsymbol{A}}^{n}_h, \underline{\boldsymbol{C}}^{k,\mathfrak{g}}_h \underline{\boldsymbol{A}}^{n}_h) \\
&\quad - \mathrm{s}^{\mathfrak{g}}_{\mathrm{div},h}(\underline{\boldsymbol{C}}^{k,\mathfrak{g}}_h \underline{\boldsymbol{A}}^{n+1}_h, \underline{\boldsymbol{C}}^{k,\mathfrak{g}}_h \underline{\boldsymbol{A}}^{n+1}_h) - \mathrm{s}^{\mathfrak{g}}_{\mathrm{div},h}(\underline{\boldsymbol{C}}^{k,\mathfrak{g}}_h \underline{\boldsymbol{A}}^{n+1}_h - \underline{\boldsymbol{C}}^{k,\mathfrak{g}}_h \underline{\boldsymbol{A}}^{n}_h, \underline{\boldsymbol{C}}^{k,\mathfrak{g}}_h \underline{\boldsymbol{A}}^{n+1}_h - \underline{\boldsymbol{C}}^{k,\mathfrak{g}}_h \underline{\boldsymbol{A}}^{n}_h) \Big),
\end{aligned}
$$

where the conclusion follows by applying the relation $b(x, y-x) = \frac{1}{2}b(y, y) - \frac{1}{2}b(x, x) - \frac{1}{2}b(y-x, y-x)$ to the symmetric bilinear forms $b = (\cdot,\cdot)_{L^2(U)\otimes\mathfrak{g}}$ and $b = \mathrm{s}^{\mathfrak{g}}_{\mathrm{div},h}$.

Finally arranging both side of the equation, moving the pure $n+1$ terms to the left and the rest to the right, we get

$$
\begin{aligned}
\frac{1}{2}\|\underline{\boldsymbol{E}}^{n+1}_h\|^2_{\mathbf{curl},\mathfrak{g},h} + \mathfrak{B}^{n+1} = \frac{1}{2}\|\underline{\boldsymbol{E}}^{n}_h\|^2_{\mathbf{curl},\mathfrak{g},h} + \mathfrak{B}^{n} \\
- \frac{1}{2}\|\underline{\boldsymbol{E}}^{n+1}_h - \underline{\boldsymbol{E}}^{n}_h\|^2_{\mathbf{curl},\mathfrak{g},h} - \frac{1}{2}\|\boldsymbol{B}^{n+1}_h - \boldsymbol{B}^{n}_h\|^2_{L^2(U)\otimes\mathfrak{g}} \\
- \frac{1}{2}\mathrm{s}^{\mathfrak{g}}_{\mathrm{div},h}(\underline{\boldsymbol{C}}^{k,\mathfrak{g}}_h \underline{\boldsymbol{A}}^{n+1}_h - \underline{\boldsymbol{C}}^{k,\mathfrak{g}}_h \underline{\boldsymbol{A}}^{n}_h, \underline{\boldsymbol{C}}^{k,\mathfrak{g}}_h \underline{\boldsymbol{A}}^{n+1}_h - \underline{\boldsymbol{C}}^{k,\mathfrak{g}}_h \underline{\boldsymbol{A}}^{n}_h),
\end{aligned}
$$

proving the statement, since the norm and $\mathrm{s}^{\mathfrak{g}}_{\mathrm{div},h}$ are both positive semidefinite. □

4. Implementation

We cover in this section the broad mechanisms of how the schemes are implemented. For our numerical simulations, this implementation was done in the `HArDCore3D` library (see `https://github.com/jdroniou/HArDCore`) starting from the serendipity DDR spaces and operators described in Section 2. This library contains a fully automated construction of these objects, including the computation of the degree depletions ℓ_{P} defined at the start of Section 2.2.

We first eliminate $\underline{\boldsymbol{A}}^{n+1}_h$ by using the first equation (3.5a) to write $\underline{\boldsymbol{A}}^{n+1}_h = \underline{\boldsymbol{A}}^{n}_h - \delta t^{n+\frac{1}{2}}\underline{\boldsymbol{E}}^{n+1}_h$. Denoting the resulting equation by $F(\boldsymbol{X}^{n+1}) = \boldsymbol{b}$, where $\boldsymbol{X}^{n+1}$ represents the combined vector of $\underline{\boldsymbol{E}}^{n+1}_h, \lambda^{n+1}_h$, we then employ the iterative Newton method to find a solution up to an accuracy of ϵ. The quantity $\underline{\boldsymbol{A}}^{n+1}_h$, which is required at the next time step, is finally recovered via back substitution.

At a single time step n, the most costly part of this process lies in the repeated assembly and resolution of the linear Newton problem: Find vectors $(\boldsymbol{X}^{n+1,i+1})_{i\in\mathbb{N}}$ such that

$$
DF_{\boldsymbol{X}^{n+1,i}}(\boldsymbol{X}^{n+1,i+1} - \boldsymbol{X}^{n+1,i}) = \boldsymbol{b} - F(\boldsymbol{X}^{n+1,i}) \quad \text{until} \quad \frac{\|F(\boldsymbol{X}^{n+1,i+1}) - \boldsymbol{b}\|_{l^2}}{\|\boldsymbol{b}\|_{l^2}} \le \epsilon.
$$

Although the derivative matrix $DF_{\boldsymbol{X}^{n+1,i}}$ is simple to determine because F is multilinear, it must be rebuilt at every iteration, with terms stemming from bilinear, trilinear, and even quadrilinear forms (see the product of bilinear brackets in the last terms of (3.6) and (3.7)). In the rest of this section, we

discuss how to perform these calculations without it becoming too expensive in either memory space or computational time.

The other major expense is resolving the linear system, for which we use the `Intel MKL PARADISO` library (see `https://software.intel.com/en-us/mkl`), which provides a multi-threaded direct solver. An efficient technique to reduce this solver cost is to apply to the linear systems the static condensation process, which eliminates all elemental unknowns of the system prior to solving. This is made possible by the specific stencil resulting from a hybrid method like (S)DDR, which couples the unknowns inside one element only with the unknowns on the faces, edges and vertices of that element (inter-element unknowns are never directly coupled). We emphasize here the difference between static condensation and the serendipity DDR process. Both reduce the number of degrees of freedom, but the SDDR spaces and operators are leaner from the start; as a result, any scheme built from it will see an improved performance in every aspect of the implementation at no additional cost. In contrast, static condensation reduces the number of unknowns only after all contributions are assembled, and thus its effect is more limited as it only reduces the cost of solving the global system, not the cost of assembling that system. Additionally, it must be repeated every iteration, with some overhead (solving a smaller linear system in each element) each time. Finally, we should highlight that, since static condensation only eliminates unknowns in the elements while serendipity also eliminates degrees of freedom on the edges/faces, the linear systems resulting from a statically condensed DDR scheme remain larger in general than the linear systems resulting from a statically condensed SDDR scheme.

4.1. LASDDR tensorisation

The numerical construction of the LASDDR complex primarily consists of wrapping a layer of matrix tensorisation around the existing SDDR code. For a code like `HArDCore3D` based on the `Eigen3` library (see `http://eigen.tuxfamily.org`), this is easily achieved using the `KroneckerProduct` functionality.

Fixing a basis $(e_I)_I$ of the d-dimensional Lie algebra $\mathfrak{g}$, each Lie algebra-valued degree of freedom can by expressed by d real values. In other words, we can think of an element of an LASDDR space as being made up of d SDDR vectors, one for each basis e_I; i.e., $\underline{\boldsymbol{v}}_h \equiv \underline{\boldsymbol{v}}_h^I \otimes e_I$. Fixing an ordering that combines everything into a single vector fixes the physical interpretation of all the remaining operators. We choose to store the values associated to each mesh entity (vertex, edge, face, element) sequentially, but for ease of distinguishing the significance of each entry, they are doubly indexed: $(\underline{\boldsymbol{v}}_h^i)^I$. The lowercase letter numbers the mesh entity it originates from, and the capital letter labels the Lie algebra basis it is attached to. As an example, if $d = 3$, we have the following structures:

$$\underline{\boldsymbol{v}}_h = \begin{bmatrix} (\underline{\boldsymbol{v}}_h^1)^1 \\ (\underline{\boldsymbol{v}}_h^1)^2 \\ (\underline{\boldsymbol{v}}_h^1)^3 \\ (\underline{\boldsymbol{v}}_h^2)^1 \\ \vdots \end{bmatrix}, \quad \underline{\boldsymbol{v}}_h^i = \begin{bmatrix} (\underline{\boldsymbol{v}}_h^i)^1 \\ (\underline{\boldsymbol{v}}_h^i)^2 \\ (\underline{\boldsymbol{v}}_h^i)^3 \end{bmatrix}, \quad \underline{\boldsymbol{v}}_h^I = \begin{bmatrix} (\underline{\boldsymbol{v}}_h^1)^I \\ (\underline{\boldsymbol{v}}_h^2)^I \\ (\underline{\boldsymbol{v}}_h^3)^I \\ (\underline{\boldsymbol{v}}_h^4)^I \\ \vdots \end{bmatrix}.$$

Then, considering a linear LASDDR operator $L^{\mathfrak{g}}$, the matrix representation $\mathbf{L}^{\mathfrak{g}}$ must by definition act as $\mathbf{L}^{\mathfrak{g}}(\underline{\boldsymbol{v}}_h) \equiv \mathbf{L}(\underline{\boldsymbol{v}}_h^I) \otimes e_I$, where $\mathbf{L}$ is the corresponding SDDR matrix operator. From some basic

arithmetic, we can conclude that $\mathbf{L}^{\mathfrak{g}}$ has the form

$$\mathbf{L}^{\mathfrak{g}} = \mathbf{L} \otimes \mathbf{I}_d = \begin{bmatrix} L_{11}\mathbf{I}_d & L_{12}\mathbf{I}_d & \cdots \\ L_{21}\mathbf{I}_d & L_{22}\mathbf{I}_d & \\ \vdots & & \ddots \end{bmatrix},$$

where $\mathbf{I}_d$ is the $d \times d$ identity matrix. For bilinear operators such as the discrete inner products $(\cdot,\cdot)_{\bullet,\mathfrak{g},h}$, and the bracket terms which we deal with in the next section, the idea is very much the same; the difference lies in the usage of a more general $d \times d$ matrix $\mathbf{M}$ in place of $\mathbf{I}_d$, indicating an interaction of the Lie algebra bases. For example, denoting the LASDDR (resp. SDDR) product matrix by $\mathbf{B}^{\mathfrak{g}}$ (resp. $\mathbf{B}$), with action previously defined as $(\underline{\boldsymbol{v}}_h)^T\mathbf{B}^{\mathfrak{g}}(\underline{\boldsymbol{w}}_h) = ((\underline{\boldsymbol{v}}_h^I)^T\mathbf{B}(\underline{\boldsymbol{w}}_h^J))\langle e_I, e_J\rangle$, the $\mathbf{M}$ becomes evidently the mass matrix of the Lie algebra:

$$\mathbf{M} = \begin{bmatrix} \langle e_1, e_1\rangle & \langle e_1, e_2\rangle & \cdots \\ \langle e_2, e_1\rangle & \langle e_2, e_2\rangle & \\ \vdots & & \ddots \end{bmatrix}, \quad \mathbf{B}^{\mathfrak{g}} = \mathbf{B} \otimes \mathbf{M} = \begin{bmatrix} B_{11}\mathbf{M} & B_{12}\mathbf{M} & \cdots \\ B_{21}\mathbf{M} & B_{22}\mathbf{M} & \\ \vdots & & \ddots \end{bmatrix}. \tag{4.1}$$

4.2. Bracket terms

The nonlinear terms in both schemes create operators which can not be encoded by a single matrix, but rather (local or global) 3 or 4-dimensional arrays:

$$\int_T \langle \boldsymbol{P}^{k,\mathfrak{g}}_{\mathbf{curl},T}\cdot, [\boldsymbol{P}^{k,\mathfrak{g}}_{\mathbf{curl},T}\cdot, P^{k+1,\mathfrak{g}}_{\mathbf{grad},T}\cdot]\rangle,$$

$$(\cdot, \star[\cdot,\cdot]^{\mathrm{div},k,h})_{\mathrm{div},\mathfrak{g},h}, \qquad (\star[\cdot,\cdot]^{\mathrm{div},k,h}, \star[\cdot,\cdot]^{\mathrm{div},k,h})_{\mathrm{div},\mathfrak{g},h},$$

$$\int_T \langle \mathbf{C}^{k,\mathfrak{g}}_T\cdot, \star[\boldsymbol{P}^{k,\mathfrak{g}}_{\mathbf{curl},T}\cdot, \boldsymbol{P}^{k,\mathfrak{g}}_{\mathbf{curl},T}\cdot]\rangle, \qquad \int_T \langle \star[\boldsymbol{P}^{k,\mathfrak{g}}_{\mathbf{curl},T}\cdot, \boldsymbol{P}^{k,\mathfrak{g}}_{\mathbf{curl},T}\cdot], \star[\boldsymbol{P}^{k,\mathfrak{g}}_{\mathbf{curl},T}\cdot, \boldsymbol{P}^{k,\mathfrak{g}}_{\mathbf{curl},T}\cdot]\rangle.$$

The tools available in `Eigen3` for dealing with these objects are not nearly as developed as the ones for matrices. In most cases, the multidimensional storage is done using the `Boost.MultiArray` library (see `https://www.boost.org/doc/libs/1_61_0/libs/multi_array/doc/index.html`), but the `Eigen::Map` function is used to interpret the data, so that we can still perform the usual matrix operations.

These generic objects are independent of time, so seemingly the most time efficient method would be to pre-compute them once and for all, and recall them when necessary. Unfortunately, the extra dimensionalities mean these operators are much larger than the (bi)linear ones appearing in LASDDR; in fact the memory usage to store these terms grows exponentially with the number of entries. The Lie algebra tensorisation only exacerbates this problem, even accounting for the many symmetries that could be exploited. This memory issue is particularly sensitive in an implementation – such as ours (which follows the HArDCore general strategy) – that assumes that each element can have its own geometry, which forces the local arrays to be computed/stored independently for each element (in a situation where the mesh elements can be classified using a few reference elements, all memory issues disappear as only local multilinear maps in reference elements need to be stored). In this context, the immense amount of memory required to store just a single one of these global maps means that there is no choice but to recalculate them at each time step and locally (mesh entity by mesh entity) as needed.

Another important effect on the runtime lies in the order in which some tensorisation-related computations are performed. Taking the sum of smaller matrices multiple times is sometimes preferable to doing it once with larger matrices (which often have lots of zeros). Therefore, delaying the tensorisation until after the vectors have been evaluated can improve both the memory usage and the speed, even though some calculations and the tensorisation have to be repeated.

We furnish these ideas with an example for the nonlinear terms of the form

$$\int_T \langle \boldsymbol{P}^{k,\mathfrak{g}}_{\mathbf{curl},T}\cdot, [\boldsymbol{P}^{k,\mathfrak{g}}_{\mathbf{curl},T}\underline{\boldsymbol{v}}_T, P^{k+1,\mathfrak{g}}_{\mathbf{grad},T}\cdot]\rangle,$$

in which $\underline{\boldsymbol{v}}_T$ is a given vector in $\underline{\boldsymbol{X}}^{k,\mathfrak{g}}_{\mathbf{curl},T}$. This term is represented by a coefficient matrix with entries (doubly indexed by $((i,I),(k,K))$) given by:

$$\int_T \langle \boldsymbol{P}^{k,\mathfrak{g}}_{\mathbf{curl},T}(\phi_i)_I, [\boldsymbol{P}^{k,\mathfrak{g}}_{\mathbf{curl},T}\underline{\boldsymbol{v}}_T, P^{k+1,\mathfrak{g}}_{\mathbf{grad},T}(\psi_k)_K]\rangle,$$

where the rows and columns range over the basis vectors, that are defined as $(\phi_i)_I \equiv \phi_i \otimes e_I$ (resp. $(\psi_k)_K$) where $(\phi_i)_i$ is a basis for $\underline{\boldsymbol{X}}^k_{\mathbf{curl},T}$ (resp. $(\psi_k)_k$ a basis of $\underline{X}^k_{\mathbf{grad},T}$). Pulling the vector out (recall that we use implicit summation), and using the definition of the L^2-product, this can be viewed as

$$\underbrace{(\underline{\boldsymbol{v}}^j_T)^J}_{\text{vector}} \underbrace{\left(\int_T \boldsymbol{P}^k_{\mathbf{curl},T}\phi_i \cdot \boldsymbol{P}^k_{\mathbf{curl},T}\phi_j\, P^{k+1}_{\mathbf{grad},T}\psi_k\right)\langle e_I, [e_J, e_K]\rangle}_{\text{trilinear form } M_{(i,I),(j,J),(k,K)}},$$

where M is indeed a trilinear form (indices $((i,I),(j,J),(k,K))$), represented by a 3-dimension block of $d^3\times(\dim(\underline{\boldsymbol{X}}^k_{\mathbf{curl},T}))\times(\dim(\underline{\boldsymbol{X}}^k_{\mathbf{curl},T}))\times(\dim(\underline{X}^k_{\mathbf{grad},T}))$ coefficients. As mentioned, although M would be useful to calculate in itself, since it can then be used to find other terms in the scheme, performing the contraction with each vector $\underline{\boldsymbol{v}}_T$ is actually quite slow because of the large matrix sums.

Instead we do something less intuitive, by combining the vector with the integral portion of the product first:

$$\underbrace{(\underline{\boldsymbol{v}}^j_T)^J \left(\int_T \boldsymbol{P}^k_{\mathbf{curl},T}\phi_i \cdot \boldsymbol{P}^k_{\mathbf{curl},T}\phi_j\, P^{k+1}_{\mathbf{grad},T}\psi_k\right)}_{M^J_{i,k}} \underbrace{\langle e_I, [e_J, e_K]\rangle}_{N_{I,J,K}}. \tag{4.2}$$

These integral coefficients represent the smaller SDDR trilinear form $\int_T(\boldsymbol{P}^k_{\mathbf{curl},T}\cdot)\cdot(\boldsymbol{P}^k_{\mathbf{curl},T}\cdot)(P^{k+1}_{\mathbf{grad},T}\cdot)$; this term is first combined (through the sum over j) with the vector $(\underline{\boldsymbol{v}}_T)^J$, before performing the tensorisation with $N_{I,J,K}$ (on $(i,I),(k,K)$), and finally taking the much shorter matrix sum over J. This delay in combining the Lie algebra indices initially seems like extra work, as this $M^J_{i,k}$ sum is unique to each vector $(\underline{\boldsymbol{v}}_T)^J$, and so the tensorisation must be re-done each time, but testing showed that the tradeoff for smaller matrix sums is worth it in this case.

The $\star[\cdot,\cdot]$ bracket is dealt with differently because it can appear twice in a single product. In this double bracket case, if we use the same summation methods as in (4.2), then we have the appearance of quadrilinear forms instead of trilinear forms, that would require the computation of a 4-dimensional array. To avoid the extra dimension, we treat the bracket terms as independent objects, that can be

manipulated separately to the product matrix. For example, in the first discretisation $\mathfrak{N}_1$ described in (3.6), two terms involve this bracket, which we compute the following way (underbrace denotes the dimension of the array representation with the implied transposition, and we write formal products to show how the calculation could be decomposed in the code):

$$(\star[\underline{\boldsymbol{v}}_h,\cdot]^{\mathrm{div},k,h},\star[\underline{\boldsymbol{w}}_h,\cdot]^{\mathrm{div},k,h})_{\mathrm{div},\mathfrak{g},h} = \underbrace{\star[\underline{\boldsymbol{v}}_h,\cdot]^{\mathrm{div},k,h}}_{\text{matrix}}\ \underbrace{(\cdot,\cdot)_{\mathrm{div},\mathfrak{g},h}}_{\text{matrix}}\ \underbrace{\star[\underline{\boldsymbol{w}}_h,\cdot]^{\mathrm{div},k,h}}_{\text{matrix}}, \tag{4.3}$$

$$(\star[\underline{\boldsymbol{v}}_h,\underline{\boldsymbol{w}}_h]^{\mathrm{div},k,h},\star[\cdot,\cdot]^{\mathrm{div},k,h})_{\mathrm{div},\mathfrak{g},h} = \underbrace{\star[\underline{\boldsymbol{v}}_h,\underline{\boldsymbol{w}}_h]^{\mathrm{div},k,h}}_{\text{vector}}\ \underbrace{(\cdot,\cdot)_{\mathrm{div},\mathfrak{g},h}}_{\text{matrix}}\ \underbrace{\star[\cdot,\cdot]^{\mathrm{div},k,h}}_{\text{3-dim array}}. \tag{4.4}$$

We note that the map $\star[\cdot,\cdot]^{\mathrm{div},k,h} : (\underline{\boldsymbol{X}}^{k,\mathfrak{g}}_{\mathbf{curl},h})^2 \to \underline{\boldsymbol{X}}^{k,\mathfrak{g}}_{\mathrm{div},h}$ is bilinear, but requires an extra dimension in the corresponding array to represent the output in $\underline{\boldsymbol{X}}^{k,\mathfrak{g}}_{\mathrm{div},h}$. Thus the terms $\star[\underline{\boldsymbol{v}}_h,\cdot]^{\mathrm{div},k,h}$ and $\star[\underline{\boldsymbol{w}}_h,\cdot]^{\mathrm{div},k,h}$ are indeed represented by matrices, calculated using the same principle (described in (4.2)) of avoiding the tensorisation until the last step. The order of operations is also crucial in (4.4); the vector-matrix multiplication must come first, to ensure that the 3-dimensional array is only contracted with a vector. We stress again however, that the full set of $\star[\cdot,\cdot]^{\mathrm{div},k,h}$ coefficients is never constructed, and all calculations are done in the way of (4.2).

The same idea is implemented in the second discretisation $\mathfrak{N}_2$ (see (3.7)), by introducing a basis for $\boldsymbol{\mathcal{P}}^{2k}(T)$, and working with the map

$$\star[\boldsymbol{P}^{k,\mathfrak{g}}_{\mathbf{curl},T}\cdot,\boldsymbol{P}^{k,\mathfrak{g}}_{\mathbf{curl},T}\cdot] : (\underline{\boldsymbol{X}}^{k,\mathfrak{g}}_{\mathbf{curl},T})^2 \to \boldsymbol{\mathcal{P}}^{2k}(T)\otimes\mathfrak{g}.$$

The numerical decomposition of the term analogous to (4.3) is

$$\underbrace{\star[\boldsymbol{P}^{k,\mathfrak{g}}_{\mathbf{curl},T}\underline{\boldsymbol{v}},\boldsymbol{P}^{k,\mathfrak{g}}_{\mathbf{curl},T}\cdot]}_{\text{matrix}}\ \underbrace{\left(\int_T\langle\cdot,\cdot\rangle\right)}_{\text{matrix}}\ \underbrace{\star[\boldsymbol{P}^{k,\mathfrak{g}}_{\mathbf{curl},T}\underline{\boldsymbol{w}},\boldsymbol{P}^{k,\mathfrak{g}}_{\mathbf{curl},T}\cdot]}_{\text{matrix}}, \tag{4.5}$$

where the integral is realised by the tensorisation of the mass matrix of the basis on $\boldsymbol{\mathcal{P}}^{2k}(T)$, and the mass matrix of the Lie algebra (see (4.1)). With orthonormal choices of bases of $\boldsymbol{\mathcal{P}}^{2k}(T)$ and $\mathfrak{g}$ (which is the default in the `HArDCore` library), the calculations here are greatly simplified; the components of $\star[\boldsymbol{P}^{k,\mathfrak{g}}_{\mathbf{curl},T}\cdot,\boldsymbol{P}^{k,\mathfrak{g}}_{\mathbf{curl},T}\cdot]$ can be found using only the triple integrals of the bases of $\mathcal{P}^k(T)$ and $\boldsymbol{\mathcal{P}}^{2k}(T)$ (without requiring to solve a linear system afterwards), and the integral in (4.5) is just given by the identity matrix

$$\mathbf{I}_{d(2k+1)} = \mathbf{I}_{2k+1}\otimes\mathbf{I}_d.$$

5. Numerical tests

We present a numerical comparison of the convergence of the two schemes, as well as the exact constraint preservation that is expected. The tests were performed on a Dell Precision 5820 desktop with a 14-core Intel Xeon processor (W-2275) clocked at 3.3 GHz and equipped with 128 GB of DDR4 RAM, running Ubuntu 22.04.1 LTS. The discretisation setting is identical to that of [2, Section 5], which only contained tests for $k = 0$ of the scheme pertaining to $\mathfrak{N}_1$. Here we expand on these results for higher orders ($k = 0, 1, 2$), and also consider the performance in relation to the second discretisation.

Let us recall the setting of these tests. The Lie algebra is $\mathfrak{g} = \mathfrak{su}(2)$, with basis

$$e_1 = -\frac{i}{2}\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad e_2 = -\frac{i}{2}\begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}, \quad e_3 = -\frac{i}{2}\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}.$$

The time interval is $[0, 1]$ and the space domain is the unit cube $(0, 1)^3$; the spatial discretisation is based on three families of Voronoi, tetrahedral, and cubic cell meshes. For each mesh of size h, the time interval is uniformly divided into $\max\{10, \lceil 5/h^{k+1} \rceil\}$ time steps; given that we use an implicit time discretisation, the expected rate of convergence is in $\delta t + h^{k+1}$, and the choice of time step is designed so that, when plotted against h, the errors should decay as h^{k+1}. To set non-zero initial conditions, and to assess the convergence properties of the schemes, we select a manufactured solution based on

$$\begin{aligned} \boldsymbol{A}(t) = &\begin{bmatrix} -0.5\cos(t)\sin(\pi x)\cos(\pi y)\cos(\pi z) \\ \cos(t)\cos(\pi x)\sin(\pi y)\cos(\pi z) \\ -0.5\cos(t)\cos(\pi x)\cos(\pi y)\sin(\pi z) \end{bmatrix} \otimes e_1 \\ &+ \begin{bmatrix} -0.5\sin(t)\sin(\pi x)\cos(\pi y)\cos(\pi z) \\ \sin(t)\cos(\pi x)\sin(\pi y)\cos(\pi z) \\ -0.5\sin(t)\cos(\pi x)\cos(\pi y)\sin(\pi z) \end{bmatrix} \otimes e_2 \\ &+ \begin{bmatrix} -0.5\sin(t)\sin^2(\pi y) \\ \cos(t)\cos^2(\pi z) \\ -0.5\sin(t)\cos^2(\pi x) \end{bmatrix} \otimes e_3, \end{aligned} \tag{5.1}$$

from which $\boldsymbol{E}(t)$ is calculated using (3.3a). Then for all tests in this section, the initial conditions are assumed to be $\underline{\boldsymbol{A}}_h^0 = \underline{\boldsymbol{I}}^{k,\mathfrak{g}}_{\mathbf{curl},h}\boldsymbol{A}(0), \underline{\boldsymbol{E}}_h^0 = \underline{\boldsymbol{I}}^{k,\mathfrak{g}}_{\mathbf{curl},h}\boldsymbol{E}(0)$.

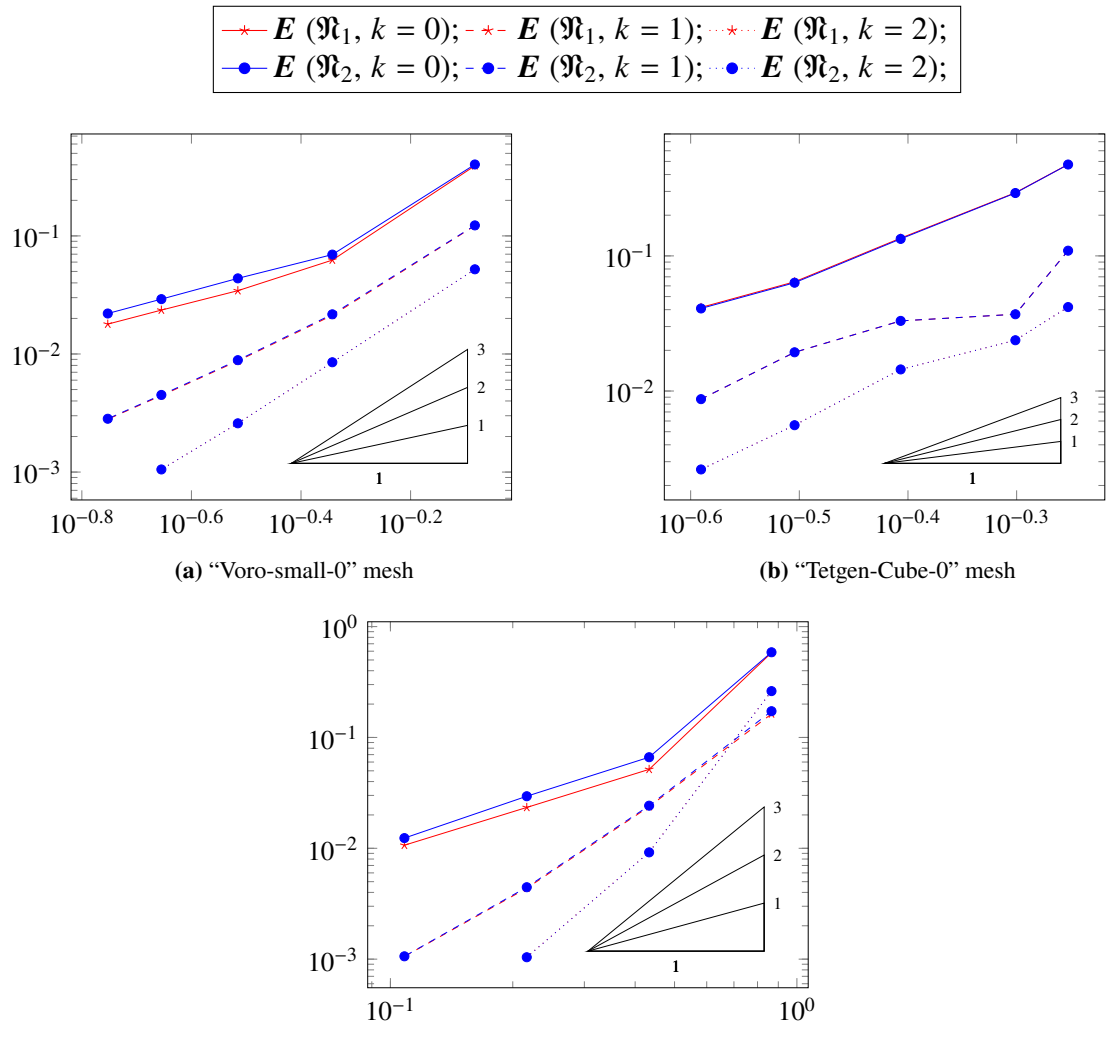
Remark 6 (Convergence results for $\underline{\lambda}_h$). Although the fields $\boldsymbol{A}$, $\boldsymbol{E}$ of a solution to the constrained formulation (3.3) solve the Yang–Mills equations, there is no proof that the λ obtained is unique. Testing performed in [2, Section 5] suggest indeed that there are infinitely many solutions; the values obtained for $\underline{\lambda}_h^n$ are therefore not very instructive, and have been omitted from the graphs. Discussion around the solvability of the linear system deriving from such a scheme can also be found in the cited section; we experienced a similar success, with a worst residual for the linear solver of the order 1e−09.

5.1. Convergence tests

In addition, appropriate boundary conditions and forcing terms are introduced to balance the equations (see [2, Section 5.1] for details). The errors for both schemes are measured by calculating the difference $\|\underline{\boldsymbol{E}}_h^{n+1} - \underline{\boldsymbol{I}}^{k,\mathfrak{g}}_{\mathbf{curl},h}\boldsymbol{E}(1)\|_{\mathbf{curl},\mathfrak{g},h}$ (resp. $\underline{\boldsymbol{A}}_h^{n+1}, \boldsymbol{A}$) at the final time, and dividing by the norm of $\underline{\boldsymbol{E}}_h^{n+1}$ (resp. $\underline{\boldsymbol{A}}_h^{n+1}$). These relative errors are plotted in Figure 1 for $\boldsymbol{E}$, and Figure 2 for $\boldsymbol{A}$.

We remark immediately that for $\boldsymbol{A}$, the errors are indistinguishable from the figure alone. The precise difference for the Voronoi and cubic sequences are calculated in Table 1, with the trend applying identically to the tetrahedral family. As either k increases or h decreases, the difference between the errors shrink accordingly, and this is seen also in the errors for $\boldsymbol{E}$ for $k = 1, 2$. However for the electric field, there is a more visible separation when $k = 0$, with the $\mathfrak{N}_1$-based discretisation performing slightly better. These tests seem to indicate that, overall, both choices $\mathfrak{N}_1, \mathfrak{N}_2$ lead to acceptable and similar results.

The schemes converge on the Voronoi and cubic meshes at the expected rate of $k+1$ for both $\boldsymbol{E}$ and $\boldsymbol{A}$, but this behaviour was not as stable for $\boldsymbol{E}$ on the tetrahedral line, where we see a rate that jumps around 3 for every k. This might be due to the asymptotic regime not been reached yet on these meshes (we note that, for $k=0$ for example, the simulation on the finest mesh seem to indicate that the convergence rate slows down). The magnitude of these errors are still ordered in the expected way, except for the coarsest cubic mesh in Figure 1, where it is corrected after the first refinement.

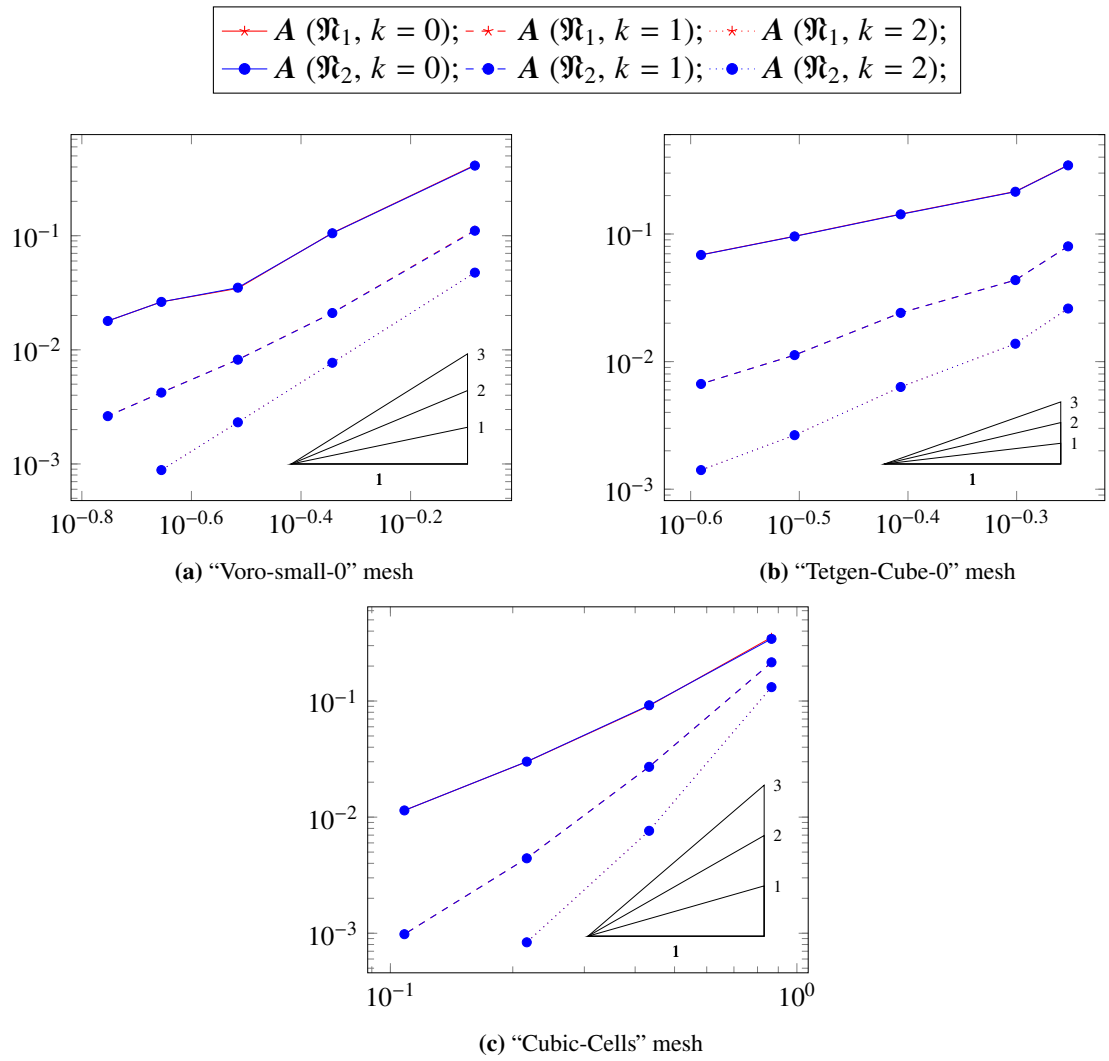


(a) "Voro-small-0" mesh

(b) "Tetgen-Cube-0" mesh

(c) "Cubic-Cells" mesh

Figure 1. Relative errors on $\boldsymbol{E}$ for $\mathfrak{N}_1$ and $\mathfrak{N}_2$: Voronoi, tetrahedral and cubic meshes.



(a) "Voro-small-0" mesh

(b) "Tetgen-Cube-0" mesh

(c) "Cubic-Cells" mesh

Figure 2. Relative errors on $\boldsymbol{A}$ for $\mathfrak{N}_1$ and $\mathfrak{N}_2$: Voronoi, tetrahedral and cubic meshes.

Table 1. Difference between the errors of $\boldsymbol{A}$ $(\mathfrak{N}_1)$ and $\boldsymbol{A}$ $(\mathfrak{N}_2)$ for various degrees k on the Voronoi and Cubic meshes.

	Voronoi mesh					Cubic mesh			
	1	2	3	4	5	1	2	3	4
$k = 0$	4.36e-3	5.98e-4	5.02e-4	1.67e-5	1.43e-5	1.05e-2	9.97e-4	8.98e-5	1.11e-5
$k = 1$	1.41e-3	8.93e-5	1.47e-5	6.23e-6	2.57e-6	9.99e-4	1.05e-4	7.04e-6	4.35e-7
$k = 2$	9.16e-5	3.46e-6	3.4e-7	6e-8	-	1.01e-4	4.97e-6	1.96e-7	-

5.2. Constraint preservation

The tests for the preservation of constraint are run with the same initial conditions as the convergence tests. For proper solutions to the Yang–Mills equations, and for Proposition 5, it is expected that these discrete fields prescribe a small or vanishing initial constraint $\mathfrak{C}^0$. This can be achieved by projecting

the initial conditions (see [2]), however for the purpose of testing Proposition 4, which does not depend on the initial values, it suffices to measure the maximum change in the functional $\mathfrak{C}^n - \mathfrak{C}^0$ over all times. The differences are presented in Table 2 for selected meshes from each sequence. We see for both methods that the constraint is stationary up to machine precision, with the small drift coming from rounding errors present at each iteration.

For reference, we also provide in Table 3 a comparison of the runtimes for the two different choices of discretisation of the nonlinear tests. The performances of both schemes are very comparable on a variety of meshes and degrees k, with the exception of a 10% difference in favour of $\mathfrak{N}_1$ for $k = 2$ on the Tetrahedral meshes, that shows up consistently through our tests. This difference is likely due to the nontrivial calculation attached to each face component (3.8a) in the definition of the discrete bracket of $\mathfrak{N}_1$. These calculations are necessary because the face values also contribute to the L^2-product, but they result in a larger dependency of the runtime on the number of faces in a particular mesh. In comparison, the $\mathfrak{N}_2$ discretisation is less affected by the shape of the elements; numerically, an extra face only represents an increase in size of the SDDR operators (which equally affects $\mathfrak{N}_1$) used in the integrals. This is supported by the results for the Voronoi sequence, that has a higher face to element ratio than the tetrahedral sequence, where $\mathfrak{N}_2$ starts to slightly outperform $\mathfrak{N}_1$.

Table 2. Maximum over n of the difference $\mathfrak{C}^n - \mathfrak{C}^0$ measured in the dual norm.

	Voronoi mesh		Tetrahedral mesh		Cubic mesh	
$\mathfrak{N}_1$	1	3	2	4	1	3
$k = 0$	8.47329e-15	3.05676e-14	2.06362e-14	4.75412e-14	3.52318e-15	2.16527e-14
$k = 1$	1.4144e-13	8.93075e-13	3.67781e-13	1.81426e-12	2.33678e-14	6.44019e-13
$k = 2$	3.69918e-12	1.18207e-10	3.82037e-12	2.77407e-11	4.45312e-14	6.48608e-12
$\mathfrak{N}_2$	1	3	2	4	1	3
$k = 0$	8.16124e-15	3.13617e-14	2.01667e-14	4.77633e-14	4.22851e-15	2.11083e-14
$k = 1$	9.8531e-14	8.83056e-13	3.69107e-13	1.81787e-12	2.52977e-14	6.48934e-13
$k = 2$	4.16428e-12	7.99416e-11	3.81537e-12	2.77391e-11	4.51672e-14	6.48285e-12

Table 3. Total runtime for each test in seconds.

	Voronoi mesh		Tetrahedral mesh		Cubic mesh	
$\mathfrak{N}_1$	1	3	2	4	1	3
$k = 0$	5.00865	145.77	4.70017	21.1378	0.564665	35.2541
$k = 1$	35.9231	2836.36	50.997	360.943	3.58296	588.679
$k = 2$	198.435	43303.9	578.499	5732.1	14.6638	14337.4
$\mathfrak{N}_2$	1	3	2	4	1	3
$k = 0$	4.57515	135.231	4.16998	19.7708	0.53204	32.879
$k = 1$	34.2036	2814.14	51.3249	340.817	3.62877	631.546
$k = 2$	190.447	42083.9	634.162	6421.87	13.8524	14414.9

6. Conclusions

We designed two schemes for the Yang–Mills equations based on the DDR method, both displaying arbitrary orders of accuracy and applications on generic polyhedral meshes. Thanks to the complex property of DDR and to the usage of a Lagrange multiplier, both schemes also preserve a discrete nonlinear constraint deriving from the Yang–Mills equations, and satisfy energy bounds. The schemes only differ in their treatment of the nonlinearity akin to a cross product combined with the Lie bracket for Lie algebra-valued vector functions. The first scheme reconstructs a discrete version of the continuous product bracket, that can then be used in the discrete L^2-products of the DDR complex. The second scheme uses the DDR potential reconstructions to get polynomials in each element, on which the continuous product bracket can be applied. We show how a clever ordering of the algebraic operations in the assembly of the schemes can help keep the computational cost at a reasonable level, despite needing to deal with multidimensional arrays and high system sizes due to the Lie algebra components. Numerical results are presented which show a good behaviour and an expected rate of convergence, with respect to the mesh size, in h^{k+1}.

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

Funded by the European Union (ERC, NEMESIS, No. 101115663). Views and opinions expressed are however those of the authors only and do not necessarily reflect those of the European Union or the European Research Council Executive Agency. Neither the European Union nor the granting authority can be held responsible for them.

Conflict of interest

The authors declare no conflicts of interest.

References

1. D. A. Di Pietro, J. Droniou, Homological- and analytical-preserving serendipity framework for polytopal complexes, with application to the DDR method, *ESAIM: M2AN*, **57** (2023), 191–225. https://doi.org/10.1051/m2an/2022067
2. J. Droniou, T. A. Oliynyk, J. J. Qian, A polyhedral discrete de rham numerical scheme for the Yang–Mills equations, *J. Comput. Phys.*, **478** (2023), 111955. https://doi.org/10.1016/j.jcp.2023.111955
3. D. N. Arnold, R. S. Falk, R. Winther, Finite element exterior calculus, homological techniques, and applications, *Acta Numer.*, **15** (2006), 1–155. https://doi.org/10.1017/S0962492906210018
4. D. Arnold, *Finite element exterior calculus*, Society for Industrial and Applied Mathematics, 2018. https://doi.org/10.1137/1.9781611975543

5. D. N. Arnold, R. S. Falk, R. Winther, Finite element exterior calculus: from Hodge theory to numerical stability, *Bull. Amer. Math. Soc.*, **47** (2010), 281–354. https://doi.org/10.1090/S0273-0979-10-01278-4
6. A. Gillette, K. Hu, S. Zhang, Nonstandard finite element de Rham complexes on cubical meshes, *Bit Numer. Math.*, **60** (2020), 373–409. https://doi.org/10.1007/s10543-019-00779-y
7. D. Arnold, K. Hu, Complexes from complexes, *Found. Comput. Math.*, **21** (2021), 1739–1774. https://doi.org/10.1007/s10208-021-09498-9
8. D. Di Pietro, M. Hanot, A discrete three-dimensional divdiv complex on polyhedral meshes with application to a mixed formulation of the biharmonic problem, *arXiv*, 2023. https://doi.org/10.48550/arXiv.2305.05729
9. L. Beirão da Veiga, F. Brezzi, L. D. Marini, A. Russo, $H(\mathrm{div})$ and $H(\mathrm{curl})$-conforming VEM, *Numer. Math.*, **133** (2016), 303–332. https://doi.org/10.1007/s00211-015-0746-1
10. L. Beirão da Veiga, F. Brezzi, F. Dassi, L. D. Marini, A. Russo, A family of three-dimensional virtual elements with applications to magnetostatics, *SIAM J. Numer. Anal.*, **56** (2018), 2940–2962. https://doi.org/10.1137/18M1169886
11. D. A. Di Pietro, J. Droniou, F. Rapetti, Fully discrete polynomial de Rham sequences of arbitrary degree on polygons and polyhedra, *Math. Models Methods Appl. Sci.*, **30** (2020), 1809–1855. https://doi.org/10.1142/S0218202520500372
12. D. A. Di Pietro, J. Droniou, An arbitrary-order discrete de Rham complex on polyhedral meshes: exactness, Poincaré inequalities, and consistency, *Found. Comput. Math.*, **23** (2023), 85–164. https://doi.org/10.1007/s10208-021-09542-8
13. D. A. Di Pietro, J. Droniou, An arbitrary-order method for magnetostatics on polyhedral meshes based on a discrete de Rham sequence, *J. Comput. Phys.*, **429** (2021), 109991. https://doi.org/10.1016/j.jcp.2020.109991
14. D. A. Di Pietro, J. Droniou, A discrete de Rham method for the Reissner-Mindlin plate bending problem on polygonal meshes, *arXiv*, 2021. https://doi.org/10.48550/arXiv.2105.11773
15. D. A. Di Pietro, J. Droniou, A fully discrete plates complex on polygonal meshes with application to the Kirchhoff–Love problem, *Math. Comp.*, **92** (2023), 51–77. https://doi.org/10.1090/mcom/3765
16. L. Chen, X. Huang, Decoupling of mixed methods based on generalized Helmholtz decompositions, *SIAM J. Numer. Anal.*, **56** (2018), 2796–2825. https://doi.org/10.1137/17M1145872
17. L. Chen, X. Huang, Finite elements for div- and divdiv-conforming symmetric tensors in arbitrary dimension, *SIAM J. Numer. Anal.*, **60** (2022), 1932–1961. https://doi.org/10.1137/21M1433708
18. L. Beirão da Veiga, F. Dassi, D. A. Di Pietro, J. Droniou, Arbitrary-order pressure-robust DDR and VEM methods for the Stokes problem on polyhedral meshes, *Comput. Meth. Appl. Mech. Eng.*, **397** (2022), 115061. https://doi.org/10.1016/j.cma.2022.115061
19. L. Beirão da Veiga, F. Dassi, G. Vacca, The stokes complex for virtual elements in three dimensions, *Math. Models Methods Appl. Sci.*, **30** (2020), 477–512. https://doi.org/10.1142/S0218202520500128
20. S. H. Christiansen, R. Winther, On constraint preservation in numerical simulations of Yang–Mills equations, *SIAM J. Sci. Comput.*, **28** (2006), 75–101. https://doi.org/10.1137/040616887

21. Y. Berchenko-Kogan, A. Stern, Charge-conserving hybrid methods for the Yang–Mills equations, *SMAI J. Comput. Math.*, **7** (2021), 97–119. https://doi.org/10.5802/smai-jcm.73

22. D. Alic, C. Bona-Casas, C. Bona, L. Rezzolla, C. Palenzuela, Conformal and covariant formulation of the Z4 system with constraint-violation damping, *Phys. Rev. D*, **85** (2012), 064040. https://doi.org/10.1103/PhysRevD.85.064040

23. O. Brodbeck, S. Frittelli, P. Hübner, O. A. Reula, Einstein's equations with asymptotically stable constraint propagation, *J. Math. Phys.*, **40** (1999), 909–923. https://doi.org/10.1063/1.532694

24. J. Frauendiener, T. Vogel, Algebraic stability analysis of constraint propagation, *Class. Quantum Grav.*, **22** (2005), 1769. https://doi.org/10.1088/0264-9381/22/9/019

25. C. Gundlach, G. Calabrese, I. Hinder, J. M. Martín-García, Constraint damping in the Z4 formulation and harmonic gauge, *Class. Quantum Grav.*, **22** (2005), 3767. https://doi.org/10.1088/0264-9381/22/17/025

26. H. Friedrich, Hyperbolic reductions for Einstein's equations, *Class. Quantum Grav.*, **13** (1996), 1451. https://doi.org/10.1088/0264-9381/13/6/014

27. A. Anderson, Y. Choquet-Bruhat, J. W. York Jr., Einstein-Bianchi hyperbolic system for general relativity, *Topol. Methods Nonlinear Anal.*, **10** (1997), 353–373. https://doi.org/10.12775/TMNA.1997.037

28. D. A. Di Pietro, J. Droniou, *The Hybrid High-Order method for polytopal meshes: design, analysis, and applications*, Springer Cham, 2020. https://doi.org/10.1007/978-3-030-37203-3

29. F. Bonaldi, D. A. Di Pietro, J. Droniou, K. Hu, An exterior calculus framework for polytopal methods, *arXiv*, 2023. https://doi.org/10.48550/arXiv.2303.11093

30. D. A. Di Pietro, J. Droniou, S. Pitassi, Cohomology of the discrete de Rham complex on domains of general topology, *Calcolo*, **60** (2023), 32. https://doi.org/10.1007/s10092-023-00523-7

31. D. A. Di Pietro, J. Droniou, A third Strang lemma and an Aubin-Nitsche trick for schemes in fully discrete formulationn, *Calcolo*, **55** (2018), 40. https://doi.org/10.1007/s10092-018-0282-3

AIMS Press