



A second-order dynamical low-rank mass-lumped finite element method for the Allen-Cahn equation

Jun Yang¹ · Nianyu Yi² · Peimeng Yin³ 

Received: 30 April 2025 / Accepted: 5 December 2025

© The Author(s), under exclusive licence to Springer Nature B.V. 2025

Abstract

In this paper, we propose a novel second-order dynamical low-rank mass-lumped finite element method for solving the Allen-Cahn equation, a semilinear parabolic partial differential equation. The matrix differential equation of the semi-discrete mass-lumped finite element scheme is decomposed into linear and nonlinear components using the second-order Strang splitting method. The linear component is solved analytically within a low-rank manifold, while the nonlinear component is discretized using a second-order augmented basis update & Galerkin integrator, in which the S -step matrix equation is solved by the explicit 2-stage strong stability-preserving Runge-Kutta method. The algorithm has lower computational complexity than the full-rank mass-lump finite element method. The dynamical low-rank finite element solution is shown to conserve mass up to a truncation tolerance at each time step for the conservative Allen-Cahn equation. Meanwhile, the modified energy is dissipative up to a high-order error and is hence stable. Numerical experiments validate the theoretical results. Symmetry-preserving tests highlight the robustness of the proposed method for long-time simulations and demonstrate its superior performance compared to existing methods.

Keywords Allen–Cahn equation · Splitting method · Dynamical low-rank approximation · Mass-lumped · Finite element method

✉ Peimeng Yin
pyin@utep.edu

Jun Yang
yangjun@smail.xtu.edu.cn

Nianyu Yi
yinianyu@xtu.edu.cn

¹ School of Mathematics and Computational Science, Xiangtan University, Xiangtan 411105, People's Republic of China

² Hunan Key Laboratory for Computation and Simulation in Science and Engineering; School of Mathematics and Computational Science, Xiangtan University, Xiangtan 411105, People's Republic of China

³ Department of Mathematical Sciences, The University of Texas at El Paso, Texas 79968, USA

Mathematics Subject Classification 35K58 · 65F55 · 65M60 · 65Y20

1 Introduction

We are interested in developing high-order numerical methods for solving gradient flows using low-dimensional surrogates that effectively capture their essential features. Dynamical low-rank approximation (DLRA) [30] is an emerging method that provides significant computational savings for solving many high-dimensional dynamical systems (e.g., [23]). Gradient flows are inherently high-dimensional but typically evolve toward an equilibrium exhibiting a low-rank structure, such as phase separations or the disappearance of patterns. In this paper, we propose a dynamical low-rank finite element method for solving the Allen-Cahn (AC) equation, a typical L^2 -gradient flow of the associated energy functional.

The AC equation [1] was originally developed to model the motion of anti-phase boundaries in crystalline solids, and the classical AC equation is given by

$$\begin{cases} w_t = \epsilon^2 \Delta w + f(w), & x \in \Omega, \quad t \in (0, T], \\ w(x, 0) = w_0, & x \in \Omega, \end{cases} \quad (1.1)$$

subject to periodic or homogeneous Neumann boundary condition, $\partial_n w = 0$, on $\partial\Omega$. For simplicity of the presentation, we take $\Omega \subset \mathbb{R}^2$ to be a convex, bounded domain. Here, w represents the concentration of one of the two metallic components of an alloy, and the parameter ϵ denotes the interfacial width, which is typically small. The nonlinear function $f(w)$ is given by $f(w) = -F'(w)$, where $F(w) = \frac{1}{4}(1 - w^2)^2$ is the double-well potential function.

The mass of solution in (1.1) is not conserved, i.e., $\frac{d}{dt} \int_{\Omega} w dx \neq 0$. A nonlocal Lagrange multiplier was introduced in [42] to construct the conservative AC equation

$$w_t = \epsilon^2 \Delta w + \tilde{f}(w), \quad (1.2)$$

where the modified nonlinear term $\tilde{f}(w)$ in (1.2) is defined as $\tilde{f}(w) := f(w) - \lambda(w)$, and $\lambda(w)$ is the Lagrange multiplier enforcing mass conservation. Two primary forms of $\lambda(w)$ have been proposed, as described by Rubinstein and Sternberg [42]:

$$\text{RSLM: } \lambda(w) = \frac{1}{|\Omega|} \int_{\Omega} f(w) dx, \quad (1.3)$$

and the nonlocal multiplier proposed by Brassel and Bretin [4]:

$$\text{BBLM: } \lambda(w) = \frac{\int_{\Omega} f(w) d\mathbf{x}}{\int_{\Omega} \sqrt{4F(w)} d\mathbf{x}} \sqrt{4F(w)}. \quad (1.4)$$

Both nonlocal Lagrange multipliers have been widely used in conservative AC equations [29, 33, 47, 52, 57]. Since the RSLM multiplier depends only on time, it has

limitations in preserving small features. Brassel and Bretin suggested that the space-time dependent Lagrange multiplier (1.4) offered better mass conservation properties.

The AC equation (1.1) can be viewed as the L^2 -gradient flow of the following Ginzburg-Landau energy functional:

$$E(w) = \int_{\Omega} \left(\frac{\epsilon^2}{2} |\nabla w|^2 + F(w) \right) d\mathbf{x}. \quad (1.5)$$

Meanwhile, the conservative AC equation (1.2) can be interpreted as the L^2 -gradient flow of the reformulated energy functional:

$$E(w) = \int_{\Omega} \left(\frac{\epsilon^2}{2} |\nabla w|^2 + F(w) + \int_0^w \lambda(s) ds \right) d\mathbf{x}. \quad (1.6)$$

The solutions of both AC equation (1.1) and its conservative AC equation (1.2) satisfy the energy dissipation law,

$$\frac{d}{dt} E(w) = \left(\frac{\delta E(w)}{\delta w}, w_t \right) = - \left\| \frac{\partial w}{\partial t} \right\|^2 \leq 0, \quad \forall t > 0, \quad (1.7)$$

where $E(w)$ is given by (1.5) for the classical AC equation (1.1), and by (1.6) for the conservative AC equation (1.2). Moreover, the conservative form (1.2) conserves mass, i.e.,

$$\frac{d}{dt} \int_{\Omega} w d\mathbf{x} = 0. \quad (1.8)$$

Preserving the physical properties, (1.7), and (1.8) in the conservative case, at the discrete level for the AC equation (1.1) and its conservative form (1.2), is a challenging and compelling research topic. Many efficient numerical methods have been developed to preserve these properties, including the invariant energy quadratization (IEQ) methods [10, 53, 54], scalar auxiliary variable (SAV) methods [22, 44, 45], splitting methods [36, 37], integrating factor Runge-Kutta (IFRK) methods [27, 28, 56], and exponential time differencing (ETD) methods [12, 14, 15, 46]. However, preserving the physical properties, even within acceptable tolerances, while reducing the computational complexity of these methods is a worthwhile endeavor.

The DLRA methods [30], which can be traced back to the Dirac–Frenkel–McLachlan variational principle from the 1930s [13, 21], have emerged as powerful tools for efficiently modeling high-dimensional systems whose dynamics evolve on lower-dimensional manifolds. More recently, the methods have been widely used in various fields, including weakly compressible flows [16], radiation transport equations [17, 20, 41], hyperbolic problems [32], advection-diffusion equations [39], and neural network applications [43]. Further reviews can be found in [18] and the references therein.

The DLRA methods evolve a dynamical system on the Riemannian manifold of fixed-rank matrices by projecting the right-hand side of a matrix differential equation onto the tangent space of the manifold, resulting in a set of differential equations that govern the factors of an SVD-type decomposition. A projector-splitting integrator for

factorized differential equations in DLRA was introduced in [38]. Later on, Ceruti and Lubich proposed a robust, *basis-update & Galerkin* (BUG) integrator, also known as an unconventional integrator [8]. This alternative integrator eliminates the backward time integration substep inherent in the projector-splitting approach, which can be unstable for dissipative problems. Additionally, the BUG integrator facilitates greater parallelism in its substeps [7]. However, the BUG integrator relies on a fixed rank, which limits its applicability when the optimal rank is either unknown or evolves dynamically during the computation. To overcome this limitation and enhance the efficiency of solving sub-differential equations in DLRA, Ceruti et al. introduced a rank-adaptive integrator [6], which mitigates unwanted projection errors in the S-step of the BUG integrator by employing augmented bases. More recently, the second-order augmented BUG integrator [5, 31] and high-order BUG integrator [40] were introduced, offering improved accuracy and robustness.

Both the finite element method (FEM) and the discontinuous Galerkin (DG) method are high-order numerical approaches offering significant advantages, including high-order accuracy on compact stencils, compatibility with *hp*-adaptivity, and flexibility in handling complex geometries. The integration of dynamical low-rank approximation (DLRA) with the DG method was explored for a homogeneous kinetic equation in [55], a solid-body rotation problem [24]. However, applying DLRA to the DG bilinear form for the Laplace term in the AC equations introduces additional complexity. The formulation on cell interfaces leads to an approximation involving a sum of four additional SVD-like matrices with differing bases, necessitating the development of specialized solvers for the resulting DLRA system. To address this complexity, we adopt FEM in the DLRA framework, where the bilinear form is approximated using a single SVD-like matrix. This approach significantly reduces the computational complexity of solving the DLRA while maintaining the high-order accuracy of FEM. Besides, it could be extended to nonuniform meshes with rectangular elements.

Integrating DLRA with numerical methods typically requires the basis functions in different directions to be orthogonal. Unlike the discontinuous Galerkin (DG) method [55], this orthogonality condition is not inherently satisfied by the FEM. To address this, we employ the mass-lumped finite element method [9], which substitutes the L^2 inner products in the variational formulation with piecewise fixed-point Gauss-Lobatto quadratures. While this approach introduces a numerical integration error relative to the original variational formulation, it does not compromise the accuracy of the finite element approximation. Moreover, the mass-lumped finite element formulation provides essential benefits for integration with DLRA: (i) The basis functions in different directions are orthogonal, making it well-suited for the integration. (ii) The coefficient matrices for polynomial-type nonlinear terms can be explicitly expressed using the Hadamard product, facilitating the low-rank approximation of the nonlinear terms.

To accurately approximate the solution to the AC problems, we primarily focus on second-order time discretization, with only a brief remark on the first-order discretization. Specifically, we employ the second-order Strang splitting method to decouple the linear and nonlinear components of the AC equations. For the linear part, we solve it analytically within a low-rank manifold. For the nonlinear part, we design a second-order augmented BUG integrator, inspired by the approach in [5]. Specifically, we will employ the explicit 2-stage strong stability preserving Runge-Kutta

method for the S-step matrix differential equation. Building on these techniques, we develop the BUG mass-lumped finite element method (BUG-MLFEM). This method employs the same formulation as the full-rank mass-lumped finite element method (FR-MLFEM) while computing solutions within low-rank subspaces of the finite element space. The computational complexity of both FR-MLFEM and BUG-MLFEM is dominated by the linear component. Specifically, FR-MLFEM has a complexity of $\mathcal{O}((m^2 + n^2) \min\{m, n\})$, whereas BUG-MLFEM reduces this to $\mathcal{O}((m^2 + n^2)r)$, where m and n denote the number of unknowns in each direction, and r is the rank of the manifold. The dynamical low-rank finite element solution is shown to conserve the mass up to a truncation tolerance at each time step. The modified energy is dissipative up to a high-order error of $\mathcal{O}(\tau(\tau^2 + h^{k+1}))$, where τ is the time step, h is the mesh size, and k is the degree of the polynomials. Consequently, the energy remains stable. Numerical examples are provided to validate the theoretical findings, with symmetry-preserving tests underscoring the robustness of the proposed method for long-time simulations.

The remainder of the paper is organized as follows. In Sect. 2, we introduce the formulation of the matrix differential equations and propose a second-order FR-MLFEM. In Sect. 3, we develop a second-order BUG-MLFEM based on the Strang-splitting method, incorporating an exact integrator for the linear component and a second-order augmented BUG integrator for the nonlinear component. Additionally, we analyze the conservation properties of the proposed method. Numerical examples are provided in Sect. 4, and concluding remarks are presented in Sect. 5.

2 Matrix differential equation

Consider the AC-type semilinear parabolic equation

$$\begin{aligned} w_t &= \epsilon^2 \Delta w + \mathcal{N}(w), & \text{in } \Omega \times (0, T), \\ w(x, t = 0) &= w_0(x), & \text{in } \Omega \times \{0\}, \\ \partial_{\mathbf{n}} w &= 0, & \text{on } \partial\Omega \times (0, T), \end{aligned} \quad (2.1)$$

where the $\mathcal{N}(w) = f(w)$ for the classical AC equation (1.1), and $\mathcal{N}(w) = \bar{f}(w)$ for the conservative AC equation (1.2).

2.1 1D semi-discrete finite element scheme

To derive the matrix ODE of (2.1), we illustrate the idea of the mass-lumped finite element method using its one-dimensional formulation. Let $\Omega = I = [a, b]$ and $a = x_1 < x_2 < \dots < x_{Mk+1} = b$ be the Gauss-Lobatto points, where $x_{ik+1}, i = 0, \dots, M$ are the nodal points. Consider a uniform mesh size $h = x_{ik+1} - x_{(i-1)k+1} = (b-a)/M$. We define the finite element space

$$V_h^k = \left\{ v \in H^1(I) : v|_{I_i} \in \mathbb{P}^k, i = 1, \dots, M \right\}, \quad (2.2)$$

where $I_i = [x_{(i-1)k+1}, x_{ik+1}]$ and \mathbb{P}^k denotes the space of polynomials of degree no more than k . Let $x_{(i-1)k+j}$ and ω_j , $j = 1, \dots, k+1$ be the quadrature points and weights of $(k+1)$ -point Gauss-Lobatto quadrature on the subinterval I_i , and denote

$$\bar{\omega}_{(i-1)k+j} = \begin{cases} \omega_j, & 2 \leq j \leq k, \\ 2\omega_j, & j = 1, k+1. \end{cases} \quad (2.3)$$

Given functions f, g , the L^2 inner product $(f, g) = \int_{\Omega} fg dx$ is approximated by the piecewise Gauss-Lobatto quadrature

$$(f, g)_h = \sum_{j=1}^{Mk+1} \bar{\omega}_j f(x_j) g(x_j). \quad (2.4)$$

The norm associated with the inner product in (2.4) is defined by

$$\|f\|_h = \sqrt{(f, f)_h}, \quad \forall f \in V_h^k.$$

Note that the $(k+1)$ -point Gauss-Lobatto quadrature is exact for polynomials of degree up to $2k-1$. Then for any $w_h, v_h \in V_h^k$, it holds

$$(\partial_x w_h, \partial_x v_h) = (\partial_x w_h, \partial_x v_h)_h. \quad (2.5)$$

The variational formulation of the AC equation (2.1) is to find $w \in L^2((0, T]; H^1(\Omega))$ such that:

$$(w_t, v) = -\epsilon^2 (\partial_x w, \partial_x v) + (\mathcal{N}(w), v), \quad \forall v \in H^1(\Omega). \quad (2.6)$$

Taking $v = v_h \in V_h^k$ and approximating the L^2 inner products in (2.6) using the Gauss-Lobatto quadrature (2.4), the variational formulation (2.6) can be formulated as

$$(w_t, v_h)_h = -\epsilon^2 (\partial_x \Pi_h w, \partial_x v_h) + (\mathcal{N}(w), v_h)_h + \mathcal{E}(v_h), \quad \forall v_h \in V_h^k, \quad (2.7)$$

$\Pi_h : C(\bar{\Omega}) \rightarrow V_h^k$ denotes the Lagrange interpolation operator at the Gauss-Lobatto points and $\mathcal{E}(v_h)$ represents both quadrature and interpolation errors satisfying [34]

$$|\mathcal{E}(v_h)| \leq C(\|\partial_t w\|_{H^{2k}} + \|w\|_{H^{k+1}(\Omega)} + \|\mathcal{N}(w)\|_{H^{2k}(\Omega)})(\|v_h\|_h + \|\partial_x v_h\|_{L^2(\Omega)})h^k.$$

The spatially semi-discrete mass-lumped finite element method for (2.1) in 1D, derived using (2.7) and (2.5), is to find $w_h \in L^2((0, T]; V_h^k)$ such that

$$(w_{h,t}, v_h)_h = -\epsilon^2 (\partial_x w_h, \partial_x v_h)_h + (\mathcal{N}(w_h), v_h)_h, \quad \forall v_h \in V_h^k, \quad (2.8a)$$

$$(w_h(x, 0), v_h) = (w_0, v_h), \quad \forall v_h \in V_h^k. \quad (2.8b)$$

2.2 Matrix differential equation in 2D

We extend the implementation using the same setting as one dimension to describe the proposed spatially semi-discrete scheme on a 2D rectangular domain $\Omega = [a, b] \times [c, d]$. To simplify the notation, we denote

$$m = Mk + 1, \quad n = Nk + 1. \quad (2.9)$$

Let $a = x_1 < x_2 < \dots < x_m = b$ represent the Gauss-Lobatto points on the interval $\Omega_x = [a, b]$, where x_{ik+1} , $i = 0, \dots, M$, are the nodal points. The uniform mesh size in the x -direction is $h_x = x_{ik+1} - x_{(i-1)k+1} = (b - a)/M$ for $i = 1, \dots, M$. Similarly, let $c = y_1 < y_2 < \dots < y_n = d$ be the Gauss-Lobatto points on the interval $\Omega_y = [c, d]$, where y_{jk+1} , $j = 0, \dots, N$, are the nodal points. The uniform mesh size in the y -direction is $h_y = y_{jk+1} - y_{(j-1)k+1} = (d - c)/N$ for $j = 1, \dots, N$. The domain $\Omega = \Omega_x \times \Omega_y$ is now divided into $M \times N$ subrectangles, collectively denoted by \mathcal{K} , using the grid points (x_{ik+1}, y_{jk+1}) , with $0 \leq i \leq M$ and $0 \leq j \leq N$. The mesh size of the partition \mathcal{K} is given by $h = \max\{h_x, h_y\}$. We define the finite element method space in each direction as:

$$\begin{aligned} V_{x,h}^k &= \left\{ \phi \in H^1(\Omega_x) : \phi|_{\Omega_{x,i}} \in \mathbb{P}^k, i = 1, \dots, M \right\}, \\ V_{y,h}^k &= \left\{ \psi \in H^1(\Omega_y) : \psi|_{\Omega_{y,j}} \in \mathbb{P}^k, j = 1, \dots, N \right\}, \end{aligned}$$

where $\Omega_{x,i} = [x_{(i-1)k+1}, x_{ik+1}]$ and $\Omega_{y,j} = [y_{(j-1)k+1}, y_{jk+1}]$. The H^1 -conforming tensor-product finite element space for Ω is given by

$$\mathcal{Q}_h^k = V_{x,h}^k \otimes V_{y,h}^k = \{v \in H^1(\Omega) : v|_K \in \mathcal{Q}_k, \quad \forall K \in \mathcal{K}\}.$$

where \mathcal{Q}_k be space of polynomials in the variables x, y with real coefficients, and of degree at most k in each direction on K . Similar to the 1D case, the L^2 inner product in 2D is approximated using the tensor product of the 1D Gauss-Lobatto quadrature, i.e., note that both f and g are separable functions, say $f(x, y) = f_1(x)f_2(y)$ and $g(x, y) = g_1(x)g_2(y)$, then the 2D L^2 inner product reduces to

$$(f, g)_h = (f_1, g_1)_h (f_2, g_2)_h,$$

where each one-dimensional inner product $(\cdot, \cdot)_h$ is defined as in (2.4). With this Gauss-Lobatto quadrature approximation, the 2D spatially semi-discrete mass-lumped finite element method for (2.2) is to find $w_h \in L^2((0, T]; \mathcal{Q}_h^k)$ such that

$$(w_{h,t}, v_h)_h = -\epsilon^2 (\nabla w_h, \nabla v_h)_h + (\mathcal{N}(w_h), v_h)_h, \quad \forall v_h \in \mathcal{Q}_h^k, \quad (2.10a)$$

$$(w_h|_{t=0}, v_h)_h = (w_0, v_h)_h, \quad \forall v_h \in \mathcal{Q}_h^k. \quad (2.10b)$$

Let $\{\phi_i(x)\}_{i=1}^m$ and $\{\psi_j(y)\}_{j=1}^n$ be basis functions for the finite elements spaces in the x - and y - directions, respectively. For any $1 \leq p \leq m$, $1 \leq q \leq n$, let x_p and y_q

represent the corresponding Gauss-Lobatto quadrature points. Then it follows

$$\phi_i(x_p) = \delta_{ip}, \quad \psi_j(y_q) = \delta_{jq}, \quad (2.11)$$

where δ_{ij} is Kronecker delta function. Moreover, for any $s > 0$, it holds

$$(\phi_i(x_p))^s = \phi_i(x_p) = \delta_{ip}, \quad \text{and} \quad (\psi_j(y_q))^s = \psi_j(y_q) = \delta_{jq}. \quad (2.12)$$

Denoted by

$$\Phi(x) = [\phi_1(x), \phi_2(x), \dots, \phi_m(x)]^\top \text{ and } \Psi(y) = [\psi_1(y), \psi_2(y), \dots, \psi_n(y)]^\top. \quad (2.13)$$

Then we can define the lumped mass matrices

$$\begin{aligned} M_x &= (\Phi(x), \Phi(x)^\top)_{\Omega_x} = [(\phi_i, \phi_j)_h]_{m \times m}, \\ M_y &= (\Psi(y), \Psi(y)^\top)_{\Omega_y} = [(\psi_i, \psi_j)_h]_{n \times n}, \end{aligned} \quad (2.14)$$

which are sparse and diagonal owing to the orthogonal property (2.11). We can similarly introduce the stiff matrices

$$\begin{aligned} A_x &= (\partial_x \Phi(x), \partial_x \Phi(x)^\top)_{\Omega_x} = [(\partial_x \phi_i, \partial_x \phi_j)_h]_{m \times m}, \\ A_y &= (\partial_y \Psi(y), \partial_y \Psi(y)^\top)_{\Omega_y} = [(\partial_y \psi_i, \partial_y \psi_j)_h]_{n \times n}, \end{aligned} \quad (2.15)$$

which are sparse, symmetric, and banded with a bandwidth $2k + 1$.

Given a function $w_h \in Q_h^k$, its expansion in terms of these bases can be written as:

$$w_h = \sum_{i=1}^m \sum_{j=1}^n W_{i,j}(t) \phi_i(x) \psi_j(y) = \Phi(x)^\top W(t) \Psi(y), \quad (2.16)$$

The matrix $W = [W_{i,j}] \in \mathbb{R}^{m \times n}$ is called the coefficient matrix of w_h , with respect to the bases $\{\phi_i(x)\}_{i=1}^m$ and $\{\psi_j(y)\}_{j=1}^n$.

Definition 2.1 Given matrices $A, B \in \mathbb{R}^{m \times n}$ with entries A_{ij} and B_{ij} , their Frobenius inner product is defined as $(A, B)_F = \text{tr}(A^\top B) = \sum_{i=1}^m \sum_{j=1}^n A_{ij} B_{ij}$. The Frobenius norm of A is $\|A\|_F = \sqrt{(A, A)_F}$. The M -weighted Frobenius inner product is $(A, B)_M = (M_x A M_y, B)_F$ with M_x and M_y given in (2.14). The M -weighted Frobenius norm of A is $\|A\|_M := \sqrt{(A, A)_M}$.

Definition 2.2 Given matrices $A, B \in \mathbb{R}^{m \times n}$ with entries A_{ij} and B_{ij} , the Hadamard product is defined as

$$A * B \in \mathbb{R}^{m \times n}, \quad \text{with entries} \quad (A * B)_{ij} = A_{ij} B_{ij}.$$

For any integer $s \geq 1$, the Hadamard power $A^{\circ s} \in \mathbb{R}^{m \times n}$ is defined as

$$A^{\circ s} = \underbrace{A * A * \cdots * A}_{s \text{ times}}, \quad \text{with entries } (A^{\circ s})_{ij} = A_{ij}^s.$$

Additionally, for $C \in \mathbb{R}^{m \times n}$, the distributive property of the Hadamard product holds:

$$A * (B + C) = A * B + A * C.$$

Then the following result holds.

Lemma 2.1 *For any $z_h, w_h \in Q_h^k$, let $Z, W \in \mathbb{R}^{m \times n}$ be their coefficient matrices, respectively. Then,*

$$(z_h, w_h)_h = (M_x Z M_y, W)_F = (Z, W)_M, \quad (2.17)$$

and for any integer $s > 0$,

$$(z_h^s, w_h)_h = (M_x Z^{\circ s} M_y, W)_F = (Z^{\circ s}, W)_M. \quad (2.18)$$

Moreover,

$$(\nabla z_h, \nabla w_h)_h = (A_x Z M_y + M_x Z A_y^\top, W)_F = -(L_x Z + Z L_y^\top, W)_M, \quad (2.19)$$

where the matrices $L_x = -M_x^{-1} A_x \in \mathbb{R}^{m \times m}$, $L_y = -M_y^{-1} A_y \in \mathbb{R}^{n \times n}$.

Proof Note that $z_h = \Phi(x)^\top Z \Psi(y)$, $w_h = \Phi(x)^\top W \Psi(y)$. Then,

$$\begin{aligned} (z_h, w_h)_h &= \left(\sum_{i=1}^m \sum_{j=1}^n Z_{ij} \phi_i(x) \psi_j(y), \sum_{v=1}^m \sum_{l=1}^n W_{vl} \phi_v(x) \psi_l(y) \right)_h \\ &= \sum_{ijvl} Z_{ij} W_{vl} (\phi_i, \phi_v)_h (\psi_j, \psi_l)_h \\ &= \sum_{vl} (M_x Z M_y^\top)_{vl} W_{vl} = (M_x Z M_y^\top, W)_F = (M_x Z M_y, W)_F. \end{aligned} \quad (2.20)$$

where M_x and M_y are defined by (2.14). For any $1 \leq p \leq m$, $1 \leq q \leq n$, let x_p and y_q represent the Gauss-Lobatto quadrature points in x - and y - directions, respectively,

$$\begin{aligned}
 (z_h^s, w_h)_h &= \left(\left(\sum_{i=1}^m \sum_{j=1}^n Z_{ij} \phi_i(x) \psi_j(y) \right)^s, \sum_{v=1}^m \sum_{l=1}^n W_{vl} \phi_v(x) \psi_l(y) \right)_h \\
 &= \sum_{p=1}^m \sum_{q=1}^n w_p w_q \left(\sum_{i=1}^m \sum_{j=1}^n Z_{ij} \phi_i(x_p) \psi_j(y_q) \right)^s \sum_{v=1}^m \sum_{l=1}^n W_{vl} \phi_v(x_p) \psi_l(y_q) \\
 &= \sum_{p=1}^m \sum_{q=1}^n w_p w_q \sum_{i=1}^m \sum_{j=1}^n Z_{ij}^s (\phi_i(x_p))^s (\psi_j(y_q))^s \sum_{v=1}^m \sum_{l=1}^n W_{vl} \phi_v(x_p) \psi_l(y_q) \\
 &= \sum_{p=1}^m \sum_{q=1}^n w_p w_q \sum_{i=1}^m \sum_{j=1}^n Z_{ij}^s \phi_i(x_p) \psi_j(y_q) \sum_{v=1}^m \sum_{l=1}^n W_{vl} \phi_v(x_p) \psi_l(y_q) \\
 &= \left(\sum_{i=1}^m \sum_{j=1}^n Z_{ij}^s \phi_i(x) \psi_j(y), \sum_{v=1}^m \sum_{l=1}^n W_{vl} \phi_v(x) \psi_l(y) \right)_h \\
 &= (M_x Z^{\text{os}} M_y, W)_F = (Z^{\text{os}}, W)_M,
 \end{aligned}$$

where (2.12) has been applied in the third and fourth equalities. Similarly,

$$\begin{aligned}
 (\nabla z_h, \nabla w_h)_h &= \left(\nabla \left(\sum_{i=1}^m \sum_{j=1}^n Z_{ij} \phi_i(x) \psi_j(y) \right), \nabla \left(\sum_{v=1}^m \sum_{l=1}^n W_{vl} \phi_v(x) \psi_l(y) \right) \right)_h \\
 &= \sum_{ijvl} Z_{ij} W_{vl} (\partial_x \phi_i, \partial_x \phi_v)_h (\psi_j, \psi_l)_h \\
 &\quad + \sum_{ijvl} Z_{ij} W_{vl} (\partial_y \psi_j, \partial_y \psi_l)_h (\phi_i, \phi_v)_h \\
 &= (A_x Z M_y, W)_F + (M_x Z A_y^\top, W)_F.
 \end{aligned}$$

where A_x and A_y are defined by (2.15). □

We introduce the coefficient matrix associated with the function $1 \in Q_h^k$, which is the all-ones matrix $\mathbf{I} = q_m q_n^\top \in \mathbb{R}^{m \times n}$, where each entry is equal to 1. Here, $q_m \in \mathbb{R}^m$ and $q_n \in \mathbb{R}^n$ are all-ones vectors.

Corollary 2.1 *For any $w_h, v_h \in Q_h^k$, let $W, V \in \mathbb{R}^{m \times n}$ be their coefficient matrices. Then,*

$$(\mathcal{N}(w_h), v_h)_h = (\mathcal{N}(W), V)_M, \quad (2.21)$$

where the nonlinear matrix operator

$$\mathcal{N}(W) := W - W^{\circ 3}, \quad (\text{for (1.1)}), \quad (2.22a)$$

$$\mathcal{N}(W) := W - W^{\circ 3} - \frac{1}{|\Omega|}(W - W^{\circ 3}, \mathbf{I}), \quad (\text{for (1.2) with RSLM}), \quad (2.22b)$$

$$\mathcal{N}(W) := W - W^{\circ 3} - \frac{(W - W^{\circ 3}, \mathbf{I}_M)}{((\mathbf{I} - W^{\circ 2}), \mathbf{I}_M)}(\mathbf{I} - W^{\circ 2}), \quad (\text{for (1.2) with BBLM}), \quad (2.22c)$$

with $\mathbf{I} \in \mathbb{R}^{m \times n}$ denoting the all-ones matrix.

Moreover, the conservative AC equation (1.2) with RSLM and BBLM holds

$$(\mathcal{N}(w_h), 1)_h = (\mathcal{N}(W), \mathbf{I}_M) = 0. \quad (2.23)$$

By Lemma 2.1 and Corollary 2.1, we have the following result.

Corollary 2.2 *Let $W \in \mathbb{R}^{m \times n}$ be the coefficient matrix of the finite element solution $w_h \in Q_h^k$, and $V \in \mathbb{R}^{m \times n}$ be the coefficient matrix of any function $v_h \in Q_h^k$. Then, the semi-discrete (2.10) is equivalent to the following problem: Find $W(t) \in \mathbb{R}^{m \times n}$ such that*

$$(\partial_t W(t), V)_M = (\mathcal{L}(W(t)), V)_M + (\mathcal{N}(W(t)), V)_M. \quad (2.24a)$$

$$W(t_0) = W_0, \quad (2.24b)$$

where $W_0 \in \mathbb{R}^{m \times n}$ is the coefficient matrix of $w_h(x, y, 0)$ in (2.10b). Here, the linear term $\mathcal{L}(W) = \epsilon^2(L_x W + W L_y^\top)$ and the nonlinear term $\mathcal{N}(W(t))$ is given by (2.22).

By the arbitrariness of V in (2.24), the 2D semi-discrete finite element scheme (2.10) can be rewritten as the matrix differential equation

$$\begin{aligned} \partial_t W &= \mathcal{L}(W(t)) + \mathcal{N}(W(t)), \\ W(t_0) &= W_0. \end{aligned} \quad (2.25)$$

2.3 Second-order FR-MLFEM

Before presenting the dynamical low-rank approximation for solving the matrix differential equation (2.25), we introduce a second order full-rank mass-lumped finite element method (FR-MLFEM).

For $n \geq 0$, let $w_h^n = w_h(x, y, t_n) \in Q_h^k$ be an approximation of $w(x, y, t_n)$, where $t_n = n\tau$ and $\tau > 0$ is a specified time step. We further denote the coefficient matrix of w_h^n by $W_n \in \mathbb{R}^{m \times n}$, namely, $w_h^n = \Phi(x)^\top W_n \Psi(y)$.

Given w_h^n or its coefficient matrix W_n , to design a second-order numerical integrator that computes W_{n+1} from the nonlinear matrix differential equation (2.25), we

consider splitting the matrix equation in (2.25) into two sub-equations associated with its linear and nonlinear terms,

$$X_t = \mathcal{L}(X), \quad (2.26a)$$

$$Z_t = \mathcal{N}(Z). \quad (2.26b)$$

We first solve the linear equation (2.26a). For any integer $\kappa > 0$ and matrix $Z \in \mathbb{R}^{\kappa \times \kappa}$, the matrix exponential is defined by

$$e^Z := \sum_{l=0}^{\infty} \frac{Z^l}{l!} = I_{\kappa} + \sum_{l=1}^{\infty} \frac{Z^l}{l!} \in \mathbb{R}^{\kappa \times \kappa}.$$

Here and throughout what follows, I_{κ} represents the identity matrix of dimension κ .

For any matrix $W \in \mathbb{R}^{m \times n}$ and $s \in \mathbb{R}$, we define the operator $e^{s\mathcal{L}}$ by

$$e^{s\mathcal{L}}W := e^{sL_x}W e^{sL_y^{\top}},$$

where L_x and L_y are defined in (2.17). It can be verified that the inverse operator of $e^{s\mathcal{L}}$ is given by $e^{-s\mathcal{L}}$, namely,

$$e^{s\mathcal{L}}e^{-s\mathcal{L}}W = e^{-s\mathcal{L}}e^{s\mathcal{L}}W = e^{-sL_x}e^{sL_x}W e^{sL_y^{\top}}e^{-sL_y^{\top}} = W.$$

The linear equation (2.26a), which a linear ordinary differential equation in matrix form, could be solved analytically, and the solution is given by

$$X_{n+1} = S_{\tau}^{\mathcal{L}}X_n = e^{\epsilon^2\tau\mathcal{L}}X_n. \quad (2.27)$$

Next, for the nonlinear problem (2.26b), we use the explicit 2-stage strong stability preserving Runge-Kutta (SSP-RK2) to approximate the solution at t_{n+1} with the initial condition Z_n by

$$\begin{cases} Z_{n,1} = Z_n + \tau\mathcal{N}(Z_n), \\ Z_{n+1} = Z_n + \frac{\tau}{2}\mathcal{N}(Z_n) + \frac{\tau}{2}\mathcal{N}(Z_{n,1}), \end{cases} \quad (2.28)$$

which can be represented in an abstract form as:

$$Z_{n+1} = S_{\tau}^{\mathcal{N}}Z_n. \quad (2.29)$$

The FR-MLFEM for (2.25) formulated in matrix form proceeds as follows: Given $W_n \in \mathbb{R}^{m \times n}$, we find $W_{n+1} \in \mathbb{R}^{m \times n}$ using a second-order Strang splitting approach, relying on the solvers outlined in (2.27) and (2.29),

$$\begin{cases} W_{n,1} = e^{\frac{\tau}{2}\epsilon^2\mathcal{L}}W_n, \\ W_{n,2} = W_{n,1} + \tau\mathcal{N}(W_{n,1}), \\ W_{n,3} = \frac{1}{2}W_{n,1} + \frac{1}{2}W_{n,2} + \frac{1}{2}\tau\mathcal{N}(W_{n,2}), \\ W_{n+1} = e^{\frac{\tau}{2}\epsilon^2\mathcal{L}}W_{n,3}, \end{cases} \quad (2.30)$$

which can be written in the abstract form:

$$W_{n+1} = S_{\frac{\tau}{2}}^{\mathcal{L}} \circ S_{\tau}^{\mathcal{N}} \circ S_{\frac{\tau}{2}}^{\mathcal{L}} W_n. \quad (2.31)$$

The FR-MLFEM (2.30) or (2.31) can be reformulated in the finite element form: Given $w_h^n = \Phi(x)^\top W_n \Psi(y) \in Q_h^k$, we find $w_h^{n+1} = \Phi(x)^\top W_{n+1} \Psi(y) \in Q_h^k$ following

$$\begin{cases} w_h^{n,1} = e^{\frac{\tau}{2}\epsilon^2\mathcal{L}_h}w_h^n, \\ (w_h^{n,2}, v_h)_h = (w_h^{n,1}, v_h)_h + \tau(\mathcal{N}(w_h^{n,1}), v_h)_h, \\ (w_h^{n,3}, v_h)_h = \frac{1}{2}(w_h^{n,1}, v_h)_h + \frac{1}{2}(w_h^{n,2}, v_h)_h + \frac{\tau}{2}(\mathcal{N}(w_h^{n,2}), v_h)_h, \\ w_h^{n+1} = e^{\frac{\tau}{2}\epsilon^2\mathcal{L}_h}w_h^{n,3}, \end{cases} \quad \forall v_h \in Q_h^k, \quad (2.32)$$

where $e^{\frac{\tau}{2}\epsilon^2\mathcal{L}_h}w_h = \Phi(x)^\top e^{\frac{\tau}{2}\epsilon^2\mathcal{L}}W\Psi(y)$, and $w_h^{n,l} = \Phi(x)^\top W_{n,l}\Psi(y) \in Q_h^k$ for $l = 1, 2, 3$.

Remark 2.1 We note that the nonlinear problem (2.28) has a computational complexity of $\mathcal{O}(mn)$, due to its explicit formulation. However, the computational cost of the FR-MLFEM scheme (2.31) or (2.32) is dominated by the two linear steps, resulting in a total complexity of $\mathcal{O}((m^2 + n^2) \min\{m, n\})$ floating-point operations.

To present the main properties of the solution obtained from the system (2.27), we introduce the following result.

Lemma 2.2 *For any $\tau > 0$, it holds*

$$(e^{\tau\epsilon^2\mathcal{L}}Z, \mathbf{I})_M = (Z, \mathbf{I})_M, \quad \forall Z \in \mathbb{R}^{m \times n}.$$

The proof of Lemma 2.2 is presented in Appendix A1. Then, we have the following results.

Lemma 2.3 *Let $W_n \in \mathbb{R}^{m \times n}$ be the solution to the system (2.30) corresponding to the conservative AC equation, and $w_h^n = \Phi(x)^\top W_n \Psi(y) \in Q_h^k$ be the finite element solution. Then the solution conserves the mass*

$$(w_h^n, 1)_h = (W_n, \mathbf{I})_M = (W_0, \mathbf{I})_M = (w_h^0, 1)_h.$$

Proof We prove this by mathematical induction. Assume that $(w_h^n, 1)_h = (w_h^0, 1)_h$. Next, we take the \mathbf{M} -weighted Frobenius inner product on both sides of (2.30) with \mathbf{I} . Using (2.23) and Lemma 2.2 yield

$$\begin{aligned}(w_h^{n+1}, 1)_h &= (W_{n+1}, \mathbf{I})_{\mathbf{M}} = (e^{\frac{\tau}{2}\epsilon^2\mathcal{L}} W_{n,3}, \mathbf{I})_{\mathbf{M}} = (W_{n,3}, \mathbf{I})_{\mathbf{M}} \\ &= \frac{1}{2}(W_{n,1}, \mathbf{I})_{\mathbf{M}} + \frac{1}{2}(W_{n,2}, \mathbf{I})_{\mathbf{M}} + \frac{1}{2}\tau(\mathcal{N}(W_{n,2}), \mathbf{I})_{\mathbf{M}} \\ &= (W_{n,1}, \mathbf{I})_{\mathbf{M}} + \frac{1}{2}\tau(\mathcal{N}(W_{n,1}), \mathbf{I})_{\mathbf{M}} \\ &= (W_{n,1}, \mathbf{I})_{\mathbf{M}} = (e^{\frac{\tau}{2}\epsilon^2\mathcal{L}} W_n, \mathbf{I})_{\mathbf{M}} = (W_n, \mathbf{I})_{\mathbf{M}} = (w_h^n, 1)_h.\end{aligned}$$

This completes the proof. \square

Similar to [36, 37], we introduce an $O(\tau)$ modification to the original energy (1.5), leading to the following modified energy

$$\tilde{E}^n = \frac{1}{2\tau}((e^{-\tau\epsilon^2\mathcal{L}_h} - 1)w_h^{n,1}, w_h^{n,1})_h + (G(w_h^{n,1}), 1)_h, \quad (2.33)$$

where $G(w_h^{n,1}) = F(0) + \int_0^{w_h^{n,1}} g(s) ds$ with

$$g(s) = -\frac{1}{2}\mathcal{N}(s) - \frac{1}{2}\mathcal{N}(s + \tau\mathcal{N}(s)). \quad (2.34)$$

We assume that there exists a constant C_g such that

$$|g'(s)| \leq C_g.$$

More discussions on the bound C_g for different types of AC equations can be found in [36, 37, 48].

Lemma 2.4 *For both the classical AC equation (1.1) and the conservative AC equation (1.2), if the time step τ satisfies $0 < \tau \leq 1/C_g$, the solution of system (2.27) or (2.32) satisfies the following energy law*

$$\tilde{E}^{n+1} \leq \tilde{E}^n.$$

The proof of Lemma 2.4 is presented in Appendix A2.

3 Dynamical low-rank mass-lumped finite element method

Let $\mathcal{M}_r \subset \mathbb{R}^{m \times n}$ be the manifold of rank- r matrices ($r \leq \min\{m, n\}$). For convenience, let $Z \in \mathbb{R}^{m \times n}$ represent the coefficient matrix of $z_h \in Q_h^k$. In the following, we use script letters to denote low-rank matrix approximations (e.g., \mathcal{Z} represents the low-rank approximation of Z). Similar to the full-rank strategy, we continue to use the

spitting (2.26) to solve the matrix differential equation (2.25) in the low-rank approximation. We begin with the low-rank approximation for the linear matrix differential equation (2.26a).

3.1 An exact low-rank integrator for the linear problem (2.26a)

Given $\tau > 0$ and a rank- r approximation $\mathcal{X}_n = U_n S_n V_n^\top$ at t_n with the factors $U_n \in \mathbb{R}^{m \times r}$ and $V_n \in \mathbb{R}^{n \times r}$ satisfying

$$U_n^\top M_x U_n = V_n^\top M_y V_n = I_r,$$

then the linear matrix differential equation (2.26a) at $t_{n+1} = t_n + \tau$ can be solved analytically by (2.27) as the rank- r matrix

$$\mathcal{X}_{n+1} = e^{\tau \epsilon^2 L_x} \mathcal{X}_n e^{\tau \epsilon^2 L_y^\top} = e^{\tau \epsilon^2 L_x} U_n S_n V_n^\top e^{\tau \epsilon^2 L_y^\top} = U_{n+1} S_{n+1} V_{n+1}^\top, \quad (3.1)$$

where the last equality is obtained by Algorithm 1, and

$$U_{n+1}^\top M_x U_{n+1} = V_{n+1}^\top M_y V_{n+1} = I_r. \quad (3.2)$$

Algorithm 1 An exact low-rank integrator to the linear matrix differential equation (2.26a).

Input: $U_n, S_n, V_n, \epsilon, \tau, M_x, M_y$.

Output: $U_{n+1}, S_{n+1}, V_{n+1}$.

- **Step 1:** Update $U^n \rightarrow U^{n+1} \in \mathbb{R}^{m \times r}$ and $V_n \rightarrow V_{n+1} \in \mathbb{R}^{n \times r}$ in parallel: perform a generalized QR (GQR) decomposition [55] with weight M_x and M_y , respectively,

$$\begin{aligned} [U_{n+1}, R] &= \text{GQR} \left(e^{\tau \epsilon^2 L_x} U_n, M_x \right), \\ [V_{n+1}, P] &= \text{GQR} \left(e^{\tau \epsilon^2 L_y} V_n, M_y \right), \end{aligned}$$

where U_{n+1} and V_{n+1} satisfy (3.2).

- **Step 2:** Update $S_n \rightarrow S_{n+1} \in \mathbb{R}^{r \times r}$ by

$$S_{n+1} = R S_n P^\top. \quad (3.3)$$

Remark 3.1 The solutions of both the FR-MLFEM scheme (2.31) and Algorithm 1 are obtained analytically. Therefore, the value of ϵ in the linear equation (2.26a) does not impose any restriction on the time step size. For the FR-MLFEM scheme (2.31), the matrix product $e^{\frac{\tau}{2} \epsilon^2 L_x} W e^{\frac{\tau}{2} \epsilon^2 L_y^\top}$ must be computed sequentially. In contrast, the matrix products $e^{\tau \epsilon^2 L_x} U_n$ and $e^{\tau \epsilon^2 L_y} V_n$ in Algorithm 1 not only have lower computational

complexity, but can also be computed in parallel. The overall computational cost of Algorithm 1 is $\mathcal{O}((m^2 + n^2)r)$.

Remark 3.2 The computational complexity of computing the exponential of an $m \times m$ matrix is $\mathcal{O}(m^3)$ in the general case [3, 25]. Several fast algorithms have been developed that reduce the complexity to $\mathcal{O}(m^\gamma)$ with $2 < \gamma \leq 3$ [26]. For more specialized cases, such as the computation of the matrix exponential for sparse matrices, the complexity is even lower [2]. In AC equation, ε is constant, so the exponential of an $m \times m$ matrix needs to be computed only once, prior to the first time step, in both the FR-MLFEM scheme (2.31) and Algorithm 1. Thus, its cost does not affect the overall computational complexity of the proposed algorithms.

Lemma 3.1 For any $\tau > 0$, given low-rank matrix $\mathcal{X}_n = U_n S_n V_n^\top$ at t_n , the low-rank approximation $\mathcal{X}_{n+1} = U_{n+1} S_{n+1} V_{n+1}^\top$, obtained in (3.1) for the linear matrix differential equation (2.26a) at t_{n+1} , satisfies (i) the rank of matrix \mathcal{X}_{n+1} is no more than that of \mathcal{X}_n , and (ii) $(U_{n+1} S_{n+1} V_{n+1}^\top, \mathbf{I})_M = (U_n S_n V_n^\top, \mathbf{I})_M$.

Proof (i) follows from (3.3). For (ii), it holds

$$\begin{aligned} (U_{n+1} S_{n+1} V_{n+1}^\top, \mathbf{I})_M &= (U_{n+1} (R S_n P^\top) V_{n+1}^\top, \mathbf{I})_M \\ &= (e^{\tau \varepsilon^2 L_x} U_n S_n (e^{\tau \varepsilon^2 L_y} V_n)^\top, \mathbf{I})_M \\ &= (U_n S_n V_n^\top, \mathbf{I})_M, \end{aligned}$$

where we have used Lemma 2.2 for the last equality. \square

3.2 DLRA for nonlinear problem (2.26b)

The Dynamical Low-Rank Approximation (DLRA) is traditionally formulated by evolving the matrix differential equation through a minimization problem using the Frobenius norm on the tangent space of \mathcal{M}_r at Z [30]. In the following, to maintain formulation equivalence between the Galerkin equation of the DLRA and the matrix variational problem (2.24), we introduce a weighted DLRA, in which the minimization problem is formulated using the M-weighted Frobenius norm in Definition 2.1. A similar weighted DLRA was investigated in [55].

Definition 3.1 The weighted DLRA to (2.26b) is defined as the solution $\mathcal{Z} \in \mathcal{M}_r$ (where \mathcal{Z} is a rank- r approximation of Z) to the differential equation

$$\partial_t \mathcal{Z} = \arg \min_{\delta \mathcal{Z} \in \mathcal{T}_Z \mathcal{M}_r} \|\delta \mathcal{Z} - \mathcal{N}(\mathcal{Z})\|_M, \quad (3.4)$$

where $\mathcal{N}(\mathcal{Z})$ is the nonlinear term defined in (2.26b), with the initial condition $\mathcal{Z}(t_n) = \mathcal{Z}_n$ being a rank- r approximation of Z_n . $\mathcal{T}_Z \mathcal{M}_r$ is the tangent space of \mathcal{M}_r at \mathcal{Z} .

Assume that matrix $\mathcal{Z} \in \mathcal{M}_r$ has a rank- r decomposition

$$\mathcal{Z} = U S V^\top, \text{ where } U^\top M_x U = V^\top M_y V = I_r, \quad (3.5)$$

with $U \in \mathbb{R}^{m \times r}$, $S \in \mathbb{R}^{r \times r}$ and $V \in \mathbb{R}^{n \times r}$. Then the tangent space of \mathcal{M}_r at \mathcal{Z} is given by [30]:

$$\mathcal{T}_{\mathcal{Z}}\mathcal{M}_r = \{\delta U S V^\top + U \delta S V^\top + U S \delta V^\top : U^\top M_x \delta U = 0, V^\top M_y \delta V = 0\}, \quad (3.6)$$

where $\delta U \in \mathbb{R}^{m \times r}$, $\delta S \in \mathbb{R}^{r \times r}$ and $\delta V \in \mathbb{R}^{n \times r}$. Under the gauge conditions $U^\top M_x \delta U = V^\top M_y \delta V = 0$, each matrix $\delta \mathcal{Z} \in \mathcal{T}_{\mathcal{Z}}\mathcal{M}_r$ has a unique decomposition

$$\begin{aligned} \delta \mathcal{Z} &= \delta U S V^\top + U \delta S V^\top + U S \delta V^\top = P_U^\perp M_x \delta \mathcal{Z} M_y P_V + P_U M_x \delta \mathcal{Z} M_y P_V \\ &\quad + P_U M_x \delta \mathcal{Z}^\top M_y P_V^\perp, \end{aligned} \quad (3.7)$$

where

$$\begin{aligned} \delta U &= P_U^\perp M_x \delta \mathcal{Z} M_y V S^{-1}, \delta S \\ &= U^\top M_x \delta \mathcal{Z} M_y V, \text{ and } \delta V = P_V^\perp M_y \delta \mathcal{Z}^\top M_x U S^{-\top}, \end{aligned} \quad (3.8)$$

with the symmetric matrices

$$\begin{aligned} P_U &= U U^\top, \quad P_U^\perp = M_x^{-1} - P_U, \\ P_V &= V V^\top, \quad P_V^\perp = M_y^{-1} - P_V. \end{aligned} \quad (3.9)$$

Here, the matrix $P_U M_x$ (resp. $P_V M_y$) is the orthogonal projection onto the column space of U (resp. V) with respect to the inner product on \mathbb{R}^m (resp. \mathbb{R}^n) with weight M_x (resp. M_y).

Similar to [30, 55], the following results hold.

Proposition 3.1 Suppose the solution $\mathcal{Z} = U S V^\top \in \mathcal{M}_r$ satisfies Definition 3.1, with initial data $\mathcal{Z}(0) = U_0 S_0 V_0^\top \in \mathcal{M}_r$ where $U_0^\top M_x U_0 = V_0^\top M_y V_0 = I_r$. Then, the following formulations are equivalent to the weighed DLRA defined in Definition 3.1.

(i) Galerkin condition: $\partial_t \mathcal{Z} \in \mathcal{T}_{\mathcal{Z}}\mathcal{M}_r$ is the solution of the Galerkin condition

$$\begin{aligned} (\partial_t \mathcal{Z} - \mathcal{N}(\mathcal{Z}), \delta \mathcal{Z})_{\mathcal{M}} &= 0, \quad \forall \delta \mathcal{Z} \in \mathcal{T}_{\mathcal{Z}}\mathcal{M}_r, \\ \mathcal{Z}(0) &= U_0 S_0 V_0^\top. \end{aligned} \quad (3.10)$$

(ii) Equations of motion: the factors of \mathcal{Z} satisfy

$$\begin{aligned} \dot{U} &= P_U^\perp M_x \mathcal{N}(\mathcal{Z}) M_y V S^{-1}, \quad \dot{S} = U^\top M_x \mathcal{N}(\mathcal{Z}) M_y V, \\ \dot{V} &= P_V^\perp M_y \mathcal{N}(\mathcal{Z})^\top M_x U S^{-\top}, \\ U(0) &= U_0, \quad S(0) = S_0, \quad V(0) = V_0, \end{aligned} \quad (3.11)$$

where P_U and P_V are defined in (3.9).

(iii) *Coupled ordinary differential equations (ODEs), referred to as the KLS system: the matrices $K = US \in \mathbb{R}^{m \times r}$, $L = VS^\top \in \mathbb{R}^{n \times r}$, and $S \in \mathbb{R}^{r \times r}$ satisfy*

$$\begin{aligned} \dot{K} &= \mathcal{N}(KV^\top)M_y V, \quad \dot{L} = \mathcal{N}(UL^\top)^\top M_x U, \quad \dot{S} = U^\top M_x \mathcal{N}(USV^\top)M_y V, \\ K(0) &= U_0 S_0, \quad L(0) = V_0 S_0^\top, \quad S(0) = S_0. \end{aligned} \quad (3.12)$$

The proof of Proposition 3.1 is similar to that of [55, Proposition 3.5].

3.3 Strategies for the nonlinear and nonlocal terms

To integrate the differential equation (3.4) in DLRA, or its equivalent formulations (3.10)–(3.12), for the nonlinear matrix differential equation (2.26b), it is essential to adopt efficient strategies for treating both the nonlinear terms and potential nonlocal terms appearing in $\mathcal{N}(W)$. In particular, the computational cost in (3.12) arises from evaluating the matrix products on the right hand side terms of the system:

$$\mathcal{N}(KV^\top)M_y V, \mathcal{N}(UL^\top)^\top M_x U, \text{ and } U^\top M_x \mathcal{N}(USV^\top)M_y V. \quad (3.13)$$

Additional overhead may result from nonlocal terms, $(W^{os}, \mathbf{I})_M$, $s = 2, 3$, as shown in (2.22).

A direct strategy (Strategy 1) is to recover the $m \times n$ matrix W in the nonlinear operator \mathcal{N} , and then compute the matrix products and inner products. The total computational complexity to compute the matrix products is $\mathcal{O}(mnr)$. Next, we consider the low-rank strategy for the nonlocal term $(W^{os}, \mathbf{I})_M$, $s = 2, 3$ in (2.22c). Recall that the all-ones matrix $\mathbf{I} = q_m q_n^\top \in \mathbb{R}^{m \times n}$, where $q_m \in \mathbb{R}^m$ and $q_n \in \mathbb{R}^n$ are all-ones vectors. We can reduce its computational complexity by using the following reformulation:

$$(W^{os}, \mathbf{I})_M = (M_x W^{os} M_y, q_m q_n^\top)_F = (M_x W^{os} M_y q_n, q_m)_F, \quad s = 2, 3. \quad (3.14)$$

Then the computational complexity in computing these nonlocal terms is $\mathcal{O}(mn)$.

An alternative strategy (Strategy 2) is to evaluate the matrix products and nonlocal terms without recovering the $m \times n$ matrix W explicitly. The coupled ODE system (3.12) implies that we can compute the product involving $\mathcal{N}(W)$ without explicitly computing $\mathcal{N}(W)$ itself. For example, in the differential equation of K , we can compute the product $\mathcal{N}(W)M_y V$ for any $V \in \mathbb{R}^{n \times r}$ without explicitly computing the elements of W . We only consider its nonlinear component $W^{o3} M_y V$ for simplicity. Recall that $K = US \in \mathbb{R}^{m \times r}$. Then, it holds

$$W = USV^\top = KV^\top = \sum_{i=1}^r K_i V_i^\top, \quad (3.15)$$

where K_i, V_i are the i -th column of K and V , respectively. In addition to the properties in Definition 2.2, the following property holds for the Hadamard product,

Table 1 Computational complexity of the nonlinear and nonlocal terms

Strategies	Criteria	Nonlinear terms in (3.13)	$(W^{\circ s}, \mathbf{I})_M$
Strategy 1	$r \geq \sqrt[3]{\min\{m, n\}}$	$\mathcal{O}(mnr)$	$\mathcal{O}(mn)$
Strategy 2	$r < \sqrt[3]{\min\{m, n\}}$	$\mathcal{O}((m+n)r^4)$	$\mathcal{O}((m+n)r^s)$

$$(K_i V_i^\top) * (K_j V_j^\top) = (K_i * K_j)(V_i * V_j)^\top, \quad \text{for } 1 \leq i, j \leq r.$$

Hence, the cubic nonlinearity which appears in the nonlinear term and $V \in \mathbb{R}^{n \times r}$ it holds:

$$\begin{aligned} (W^{\circ 3})M_y V &= \left(\left(\sum_{i=1}^r K_i V_i^\top \right) * \left(\sum_{j=1}^r K_j V_j^\top \right) * \left(\sum_{l=1}^r K_l V_l^\top \right) \right) M_y V \\ &= \left(\sum_{i,j,l=1}^r (K_i * K_j * K_l) (V_i * V_j * V_l)^\top \right) M_y V \\ &= \sum_{i,j,l=1}^r (K_i * K_j * K_l) \left((V_i * V_j * V_l)^\top M_y V \right). \end{aligned} \quad (3.16)$$

Therefore, the computation of $(W^{\circ 3})M_y V$, when implemented following (3.16), results in a computational complexity of $\mathcal{O}((m+n)r^4)$. Analogously, the computational complexity of $(W^{\circ 3})^\top M_x U$ with $U \in \mathbb{R}^{m \times r}$ is also $\mathcal{O}((m+n)r^4)$. By (3.16), the dominant term in the product $U^\top M_x \mathcal{N}(U S V^\top) M_y V$ has the following reformulation

$$U^\top M_x (W^{\circ 3}) M_y V = \sum_{i,j,l=1}^r \left(U^\top M_x (K_i * K_j * K_l) \right) \left((V_i * V_j * V_l)^\top M_y V \right), \quad (3.17)$$

which has the computational complexity $\mathcal{O}((m+n)r^4)$.

Since M_x is a diagonal matrix, the computational complexity of computing $M_x W^{\circ 3} M_y q_n$, which dominates the computational complexity of $(W^{\circ 3}, \mathbf{I})_M$, is equivalent to that of computing $W^{\circ 3} M_y q_n$. The decomposition of $W^{\circ 3} M_y q_n$ follows the same as (3.16), with V replaced by q_n , and its computational complexity is $\mathcal{O}((m+n)r^3)$, which is also the computational complexity of $(W^{\circ 3}, \mathbf{I})_M$. Similarly, the computational complexity for the nonlocal term $(W^{\circ 2}, \mathbf{I})_M$ in (2.22c) is $\mathcal{O}((m+n)r^2)$.

We summarize the computational complexity of the nonlinear and nonlocal terms for Strategy 1 and Strategy 2 in Table 1. In typical applications, the relation $r^2 \leq \min\{m, n\}$ usually holds. Based on this, we adopt the following guideline for selecting the strategy: if $r < \sqrt[3]{\min\{m, n\}}$, we choose Strategy 2; otherwise, we choose Strategy 1. Following this principle, the computational complexity for solving the nonlinear problem (2.26b) is bounded by $\mathcal{O}(mnr)$.

3.4 Second-order integrator

Similar to the Strang splitting approach for the FR-MLFEM (2.31), we propose a second-order low-rank integrator for the matrix differential equation (2.25) by solving the split linear and nonlinear matrix differential equations (2.26a) and (2.26b). For the linear component (2.26a), we use the exact low-rank integrator in Algorithm 1.

For the nonlinear component (2.26b), we employ the augmented BUG integrator [5, 6] for the KLS system (3.12), an equivalent formulation of DLRA to the nonlinear matrix differential equation (2.26b). This method extends the bases used in the S-step of the BUG integrator [6, 7] to eliminate unwanted projection errors. The BUG integrator, viewed as a splitting method, is applied to the KLS system (3.12), where the K and L equations are updated in parallel, followed by an update of the S equation.

The second-order augmented BUG integrator [5] builds upon the first-order augmented BUG integrator [8]. We start by outlining the key aspects of the first-order integrator for the matrix differential equation (2.26b). Given a factored rank- r matrix $\mathcal{Z}_n = U_n S_n V_n^\top$ at t_n , with factors $U_n \in \mathbb{R}^{m \times r}$ and $V_n \in \mathbb{R}^{n \times r}$ satisfying

$$U_n^\top M_x U_n = V_n^\top M_y V_n = I_r,$$

the one-step first-order augmented BUG integrator gives a low-rank approximation at the next time step $t_{n+1} = t_n + \tau$, namely,

$$\mathcal{Z}_{n+1} = U_{n+1} S_{n+1} V_{n+1}^\top, \quad (3.18)$$

where the updated bases $U_{n+1} \in \mathbb{R}^{m \times \hat{r}}$ and $V_{n+1} \in \mathbb{R}^{n \times \hat{r}}$ satisfy

$$U_{n+1}^\top M_x U_{n+1} = V_{n+1}^\top M_y V_{n+1} = I_{\hat{r}}, \quad \text{where } \hat{r} \leq 2r.$$

The specifics of the first-order augmented BUG integrator are outlined in Algorithm 2.

Algorithm 2 One-step first-order augmented BUG integrator for (2.26b).

Input: $U_n, S_n, V_n, \tau, M_x, M_y$.

Output: $U_{n+1}, S_{n+1}, V_{n+1}$.

- **Step 1:** Update $U_n \rightarrow U_{n+1} \in \mathbb{R}^{m \times \hat{r}}$ and $V_n \rightarrow V_{n+1} \in \mathbb{R}^{n \times \hat{r}}$ in parallel, with $(\hat{r} \leq 2r)$:

- K -step:

- Solve K_{n+1} from the $m \times r$ matrix equation

$$\frac{K_{n+1} - K_n}{\tau} = \mathcal{N}(K_n V_n^\top) M_y V_n, \quad K_n = U_n S_n. \quad (3.19)$$

- L -step:

- Solve L_{n+1} from the $n \times r$ matrix equation

$$\frac{L_{n+1} - L_n}{\tau} = \mathcal{N}(U_n L_n^\top)^\top M_x U_n, \quad L_n = V_n S_n^\top. \quad (3.20)$$

– Update the basis:

- Set $\tilde{K}_{n+1} = [K_{n+1}, U_n]$ and $\tilde{L}_{n+1} = [L_{n+1}, V_n]$.
- Compute $U_{n+1} \in \mathbb{R}^{m \times \hat{r}}$, $V_{n+1} \in \mathbb{R}^{n \times \hat{r}}$ following generalizing QR (GQR) decomposition in [55],

$$[U_{n+1}, \sim] = \text{GQR}(\tilde{K}_{n+1}, M_x), \quad [V_{n+1}, \sim] = \text{GQR}(\tilde{L}_{n+1}, M_y).$$

- Compute the matrix $M = U_{n+1}^\top M_x U_n$, and $N = V_{n+1}^\top M_y V_n$.
 - **Step 2:** Update $S_n \rightarrow S_{n+1} \in \mathbb{R}^{\hat{r} \times \hat{r}}$:
- S-step:

- Set $S_{n,*} = M S_n N^\top$.
- Solve the $\hat{r} \times \hat{r}$ matrix equation

$$\frac{S_{n+1} - S_{n,*}}{\tau} = U_{n+1}^\top M_x \mathcal{N}(U_{n+1} S_{n,*} V_{n+1}^\top) M_y V_{n+1}. \quad (3.21)$$

Remark 3.3 Since the first-order augmented BUG integrator in Algorithm 2 is a component of the second-order integrator for the nonlinear matrix differential equation (2.26b), the truncation step is deferred to the main algorithm for the matrix differential equation (2.26b).

With the preparation above, we present the second-order integrator for the matrix differential equations in (2.26). The exact low-rank integrator from Algorithm 1 is employed for the linear matrix differential equation (2.26a), while the second-order augmented BUG integrator [5] is applied to solve the nonlinear matrix differential equation (2.26b). In particular, we will use SSP-RK2 to solve the matrix equation in the S-step. This approach utilizes a second-order Strang splitting method, similar to (2.31) or (2.30), as implemented in FR-MLFEM, to integrate the linear and nonlinear integrators. The details are outlined in Algorithm 3.

Algorithm 3 A second-order integrator for the split matrix differential equations in (2.26).

Input: $U_n, S_n, V_n, \tau, M_x, M_y$.

Output: $U_{n+1}, S_{n+1}, V_{n+1}$.

- **Linear Step:** Perform Algorithm 1 with a time step $\frac{\tau}{2}$ and initial values U_n, S_n, V_n to obtain

$$\mathcal{W}_{n,1} = U_{n,1} S_{n,1} V_{n,1}^\top.$$

• **Nonlinear Step:**

– Trapezoidal Approximation:

- Perform Algorithm 2 with a time step τ and initial values $U_{n,1}$, $S_{n,1}$, $V_{n,1}$ to obtain the approximation of rank $\hat{r} = 2r$,

$$\mathcal{W}_{n,2} = U_{n,2} S_{n,2} V_{n,2}^\top. \quad (3.22)$$

– Galerkin Step:

- Assemble the augmented matrices \mathbb{U} and \mathbb{V} ,

$$\begin{aligned} \mathbb{U} &= [q_m, U_{n,1}, \tau \mathcal{N}(\mathcal{W}_{n,1}) M_y V_{n,1}, \tau \mathcal{N}(\mathcal{W}_{n,2}) M_y V_{n,2}], \\ \mathbb{V} &= [q_n, V_{n,1}, \tau \mathcal{N}(\mathcal{W}_{n,1})^\top M_x U_{n,1}, \tau \mathcal{N}(\mathcal{W}_{n,2})^\top M_x U_{n,2}]. \end{aligned}$$

- Compute the new bases:

$$[\bar{U}_{n,3}, \sim] = \text{GQR}(\mathbb{U}, M_x), \quad [\bar{V}_{n,3}, \sim] = \text{GQR}(\mathbb{V}, M_y). \quad (3.23)$$

- Compute the $\bar{r} \times r$ matrices $M = \bar{U}_{n,3}^\top M_x U_{n,1}$ and $N = \bar{V}_{n,3}^\top M_y V_{n,1}$, and $\bar{r} \times \bar{r}$ matrix

$$\bar{S}_{n,1} = M S_{n,1} N^\top. \quad (3.24)$$

- Solve the $\bar{r} \times \bar{r}$ matrix equations by the SSP-RK2:

$$\bar{S}_{n,2} = \bar{S}_{n,1} + \tau \bar{U}_{n,3}^\top M_x \mathcal{N}(\bar{U}_{n,3} \bar{S}_{n,1} \bar{V}_{n,3}^\top) M_y \bar{V}_{n,3}, \quad (3.25a)$$

$$\bar{S}_{n,3} = \frac{1}{2} \bar{S}_{n,1} + \frac{1}{2} \bar{S}_{n,2} + \frac{\tau}{2} \bar{U}_{n,3}^\top M_x \mathcal{N}(\bar{U}_{n,3} \bar{S}_{n,2} \bar{V}_{n,3}^\top) M_y \bar{V}_{n,3}. \quad (3.25b)$$

– Truncation Step:

- Compute the SVD decomposition of $\bar{S}_{n,3} = \bar{R} \bar{\Sigma} \bar{P}^\top$ with $\bar{\Sigma} = \text{diag}(\sigma_i)$.
- Truncate $\bar{\Sigma}$ to $S_{n,3}$ with either the original rank r or a new rank $r = \tilde{r}$, determined by prescribed a truncation tolerance η , such that

$$\left(\sum_{i=r+1}^{\tilde{r}} \sigma_i^2 \right)^{\frac{1}{2}} \leq \eta. \quad (3.26)$$

- Set $R \in \mathbb{R}^{\tilde{r} \times r}$ and $P \in \mathbb{R}^{\tilde{r} \times r}$ containing the first r columns of \bar{R} and \bar{P} , respectively.

- Set $U_{n,3} = \overline{U}_{n,3}R \in \mathbb{R}^{m \times r}$ and $V_{n,3} = \overline{V}_{n,3}P \in \mathbb{R}^{n \times r}$.
- **Linear Step:** Perform Algorithm 1 with a time step $\frac{\tau}{2}$ and initial values $U_{n,3}, S_{n,3}, V_{n,3}$ to obtain

$$\mathcal{W}_{n+1} = U_{n+1} S_{n+1} V_{n+1}^\top.$$

Remark 3.4 The nonlinear step in Algorithm 3 has a computational complexity of at most $\mathcal{O}(mnr)$. Since the linear steps remain the dominant cost in the second-order integrator Algorithm 3, the overall computational complexity is $\mathcal{O}((m^2 + n^2)r)$.

Remark 3.5 In practice, the GQR method in Algorithm 2 and Algorithm 3 can be replaced with the RGQR method (Algorithm 4), which substantially improves computational efficiency while maintaining accuracy.

Remark 3.6 A first-order integrator can also be proposed by using the Lie-Trotter splitting [49], which combines Algorithm 1 and Algorithm 2 analogous to Algorithm 3. We omit the details.

3.5 Properties of the dynamical low-rank finite element solution

In this section, we explore the properties of the solution from Algorithm 3. Recall that script letters denote low-rank matrix approximations (e.g., \mathcal{Z} represents the low-rank approximation of the matrix Z). From this subsection, we adopt a bold italic font, such as $\mathbf{z}_h = \Phi(x)^\top \mathcal{Z} \Psi(y) \in \mathcal{Q}_h^k$, to represent the low-rank function associated with the low-rank matrix \mathcal{Z} . First, we defined the following subspaces of the finite element space \mathcal{Q}_h^k ,

$$\mathcal{Q}_0^{n,k} = \left\{ v \mid v(x, y) = \Phi(x)^\top U_n S V_n^\top \Psi(y), \quad \forall S \in \mathbb{R}^{r \times r} \right\}, \quad (3.27a)$$

$$\mathcal{Q}_1^{n,k} = \left\{ v \mid v(x, y) = \Phi(x)^\top U_{n,1} S V_{n,1}^\top \Psi(y), \quad \forall S \in \mathbb{R}^{r \times r} \right\}, \quad (3.27b)$$

$$\overline{\mathcal{Q}}_3^{n,k} = \left\{ v \mid v(x, y) = \Phi(x)^\top \overline{U}_{n,3} S \overline{V}_{n,3}^\top \Psi(y), \quad \forall S \in \mathbb{R}^{\bar{r} \times \bar{r}} \right\}, \quad (3.27c)$$

$$\mathcal{Q}_3^{n,k} = \left\{ v \mid v(x, y) = \Phi(x)^\top U_{n,3} S V_{n,3}^\top \Psi(y), \quad \forall S \in \mathbb{R}^{\tilde{r} \times \tilde{r}} \right\}, \quad (3.27d)$$

where r , \bar{r} , and \tilde{r} are given in Algorithm 3.

Remark 3.7 We have used the adaptive rank \tilde{r} for S matrix in the subspace $\mathcal{Q}_3^{n,k}$ in (3.27d), and it can be changed to fixed rank r as stated in Truncation Step in Algorithm 3.

By (3.23) in Algorithm 3, it can be observed that

$$\mathcal{Q}_1^{n,k} \subset \overline{\mathcal{Q}}_3^{n,k}. \quad (3.28)$$

Similar to the full-rank finite element method, Algorithm 3 provides a second-order low-rank finite element approximation for the matrix differential equations in (2.26). Notably, the low-rank finite element formulations in Algorithm 3 are mathematically equivalent to the full-rank finite element scheme (2.32), differing only in that the approximations are computed in subspaces of the original finite element space. Specifically, the following statement holds.

Lemma 3.2 *Algorithm 3 can be reformulated as the BUG-MLFEM: Given $\mathbf{w}_h^n = \Phi(x)^\top U_n S_n V_n^\top \Phi(y) \in Q_0^{n,k}$, we find $\mathbf{w}_h^{n+1} = \Phi(x)^\top U_{n+1} S_{n+1} V_{n+1}^\top \Phi(y) \in Q_0^{n+1,k}$ such that*

$$\mathbf{w}_S^{n,1} = e^{\frac{\tau}{2}\epsilon^2 \mathcal{L}_h} \mathbf{w}_h^n \quad (3.29a)$$

$$(\mathbf{w}_S^{n,2}, \mathbf{w}_2)_h = (\mathbf{w}_S^{n,1}, \mathbf{w}_2)_h + \tau(\mathcal{N}(\mathbf{w}_S^{n,1}), \mathbf{w}_2)_h, \quad \forall \mathbf{w}_2 \in \overline{Q}_3^{n,k}, \quad (3.29b)$$

$$(\mathbf{w}_S^{n,3}, \mathbf{w}_3)_h = \frac{1}{2}(\mathbf{w}_S^{n,1}, \mathbf{w}_3)_h + \frac{1}{2}(\mathbf{w}_S^{n,2}, \mathbf{w}_3)_h + \frac{\tau}{2}(\mathcal{N}(\mathbf{w}_S^{n,2}), \mathbf{w}_3)_h, \quad \forall \mathbf{w}_3 \in \overline{Q}_3^{n,k}, \quad (3.29c)$$

$$\mathbf{w}_h^{n+1} = e^{\frac{\tau}{2}\epsilon^2 \mathcal{L}_h} \mathbf{w}_S^{n,3}. \quad (3.29d)$$

Here, the operator $e^{\frac{\tau}{2}\epsilon^2 \mathcal{L}_h}$ is defined in (2.32), and the low-rank finite element approximations

$$\begin{aligned} \mathbf{w}_S^{n,1} &= \Phi(x)^\top U_{n,1} S_{n,1} V_{n,1}^\top \Phi(y) \in Q_1^{n,k}, \\ \mathbf{w}_S^{n,2} &= \Phi(x)^\top \overline{U}_{n,3} \overline{S}_{n,2} \overline{V}_{n,3}^\top \Phi(y) \in \overline{Q}_3^{n,k}, \\ \mathbf{w}_S^{n,3} &= \Phi(x)^\top \overline{U}_{n,3} \overline{S}_{n,3} \overline{V}_{n,3}^\top \Phi(y) \in \overline{Q}_3^{n,k}, \\ \mathbf{w}_S^{n,3} &= \Phi(x)^\top U_{n,3} S_{n,3} V_{n,3}^\top \Phi(y) \in Q_3^{n,k}, \end{aligned}$$

where $\mathbf{w}_S^{n,3}$ is the truncated low-rank finite element approximations of $\mathbf{w}_S^{n,3}$.

Proof It can be verified that the (3.29a) and (3.29d) are equivalent to the two linear steps in Algorithm 3, respectively. Below, we demonstrate the equivalence for (3.29a); the proof for (3.29d) follows a similar argument. By Algorithm 1, it holds

$$\begin{aligned} \mathbf{w}_S^{n,1} &= \Phi(x)^\top U_{n,1} S_{n,1} V_{n,1}^\top \Phi(y) = \Phi(x)^\top U_{n,1} R S_n P^\top V_{n,1}^\top \Phi(y) \\ &= \Phi(x)^\top e^{\frac{\tau}{2}\epsilon^2 L_x} U_n S_n V_n^\top e^{\frac{\tau}{2}\epsilon^2 L_y} \Phi(y) = e^{\frac{\tau}{2}\epsilon^2 \mathcal{L}_h} \mathbf{w}_h^n. \end{aligned} \quad (3.30)$$

In this following, we focus on (3.29b) and (3.29c). Let

$$\mathbf{w}_S^{n,1} = \Phi(x)^\top \overline{U}_{n,3} \overline{S}_{n,1} \overline{V}_{n,3}^\top \Psi(y) \in \overline{Q}_3^{n,k},$$

be the projection of $\mathbf{w}_S^{n,1} \in Q_1^{n,k}$ onto $\overline{Q}_3^{n,k}$, namely,

$$(\mathbf{w}_S^{n,1}, \mathbf{w}_h)_h = (\mathbf{w}_S^{n,1}, \mathbf{w}_h)_h, \quad \forall \mathbf{w}_h \in \overline{Q}_3^{n,k},$$

which, by Lemma 2.1, is equivalent to the matrix equation

$$\begin{aligned} & \left(M_x \bar{U}_{n,3} \bar{S}_{n,1} \bar{V}_{n,3}^\top M_y, \bar{U}_{n,3} S \bar{V}_{n,3}^\top \right)_F \\ &= \left(M_x \bar{U}_{n,3} S_{n,1} \bar{V}_{n,3}^\top M_y, \bar{U}_{n,3} S \bar{V}_{n,3}^\top \right)_F, \quad \forall S \in \mathbb{R}^{\bar{r} \times \bar{r}}. \end{aligned} \quad (3.31)$$

From (3.28), it also holds

$$\mathbf{w}_S^{n,1} = \mathbf{w}_S^{n,1} \in \bar{Q}_3^{n,k},$$

By Lemma 2.1 and Corollary 2.1, the low-rank finite element formulation (3.29b) is equivalent to

$$\begin{aligned} & \left(M_x \bar{U}_{n,3} (\bar{S}_{n,2} - S_{n,1}) \bar{V}_{n,3}^\top M_y, \bar{U}_{n,3} S \bar{V}_{n,3}^\top \right)_F \\ &= \tau \left(M_x \mathcal{N}(U_{n,1} S_{n,1} V_{n,1}^\top) M_y, \bar{U}_{n,3} S \bar{V}_{n,3}^\top \right)_F, \quad \forall S \in \mathbb{R}^{\bar{r} \times \bar{r}}. \end{aligned}$$

Using (3.31) and the properties of the Frobenius inner product gives

$$(\bar{S}_{n,2}, S)_F = (\bar{S}_{n,1}, S)_F + \tau \left(\bar{U}_{n,3}^\top M_x \mathcal{N}(U_{n,1} S_{n,1} V_{n,1}^\top) M_y \bar{V}_{n,3}, S \right)_F, \quad \forall S \in \mathbb{R}^{\bar{r} \times \bar{r}}. \quad (3.32)$$

From (3.23), $\bar{U}_{n,3} \bar{U}_{n,3}^\top M_x$ is the orthogonal projection onto the range of $\bar{U}_{n,3}$, which by definition equals the range of \bar{U} . In particular, the columns of $U_{n,1}$ lie in range of $\bar{U}_{n,3}$, and hence $\bar{U}_{n,3} \bar{U}_{n,3}^\top M_x U_{n,1} = \bar{U}_{n,3} M = U_{n,1}$. Similarly, $\bar{V}_{n,3} \bar{V}_{n,3}^\top M_y V_{n,1} = \bar{V}_{n,3} N = V_{n,1}$. Here, M and N were given in Algorithm 3. Therefore, it holds

$$U_{n,1} S_{n,1} V_{n,1}^\top = \bar{U}_{n,3} M S_{n,1} N^\top \bar{V}_{n,3}^\top = \bar{U}_{n,3} \bar{S}_{n,1} \bar{V}_{n,3}^\top. \quad (3.33)$$

Then, (3.32) can be written as

$$(\bar{S}_{n,2}, S)_F = (\bar{S}_{n,1}, S)_F + \tau \left(\bar{U}_{n,3}^\top M_x \mathcal{N}(\bar{U}_{n,3} \bar{S}_{n,1} \bar{V}_{n,3}^\top) M_y \bar{V}_{n,3}, S \right)_F, \quad \forall S \in \mathbb{R}^{\bar{r} \times \bar{r}},$$

which gives (3.25a) by the arbitrariness of S . Similarly, (3.29c) is equivalent to (3.25b). \square

Remark 3.8 In Lemma 3.2, we presented the finite element formulations for the S -step but omitted those for the K -step and L -step, as the finite element solution for the S -step is equivalent to the low-rank finite element solution.

Theorem 3.1 Given $\mathbf{w}_h^n \in Q_0^{n,k}$, the BUG-MLFEM solution $\mathbf{w}_h^{n+1} \in Q_0^{n+1,k}$ obtained from Algorithm 3 for the conservative AC equation (1.2) conserves the mass up to the truncation the tolerance η specified in (3.26), i.e.,

$$\begin{aligned} & |(\mathbf{w}_h^{n+1} - \mathbf{w}_h^n, \mathbf{1})_h| = |(\mathcal{W}_{n+1}, \mathbf{I})_M - (\mathcal{W}_n, \mathbf{I})_M| \\ &= |(U_{n+1} S_{n+1} V_{n+1}^\top, \mathbf{I})_M - (U_n S_n V_n^\top, \mathbf{I})_M| \leq C \eta. \end{aligned}$$

Proof The matrices $U_{n,1}$, $S_{n,1}$, $V_{n,1}$ are obtained from Algorithm 1 with initial values $\{U_n, S_n, V_n\}$ and the time step $\frac{\tau}{2}$. Then, by Lemma 3.1, it follows

$$(U_{n,1} S_{n,1} V_{n,1}^\top, \mathbf{I})_M = (U_n S_n V_n, \mathbf{I})_M. \quad (3.34)$$

Recall that the all-ones matrix $\mathbf{I} = q_m q_n^\top \in \mathbb{R}^{m \times n}$, where $q_m \in \mathbb{R}^m$ and $q_n \in \mathbb{R}^n$ are all-ones vectors. Similar to (3.33), $\bar{U}_{n,3} \bar{U}_{n,3}^\top M_x q_m = q_m$, and $\bar{V}_{n,3} \bar{V}_{n,3}^\top M_y q_n = q_n$. Then, for any matrix $W \in \mathbb{R}^{m \times n}$,

$$\begin{aligned} \left(\bar{U}_{n,3} \bar{U}_{n,3}^\top M_x \mathcal{N}(W) M_y \bar{V}_{n,3} \bar{V}_{n,3}^\top, \mathbf{I} \right)_M &= \left(M_x \bar{U}_{n,3} \bar{U}_{n,3}^\top M_x \mathcal{N}(W) M_y \bar{V}_{n,3} \bar{V}_{n,3}^\top M_y, \mathbf{I} \right)_F \\ &= (M_x \mathcal{N}(W) M_y, \mathbf{I})_F = (\mathcal{N}(W), \mathbf{I})_M. \end{aligned} \quad (3.35)$$

Therefore, $\bar{U}_{n,3}$, $\bar{S}_{n,3}$, $\bar{V}_{n,3}$ obtained from Algorithm 3 with initial values $U_{n,1}$, $S_{n,1}$, $V_{n,1}$ satisfy

$$\begin{aligned} &(\bar{U}_{n,3} \bar{S}_{n,3} \bar{V}_{n,3}^\top, \mathbf{I})_M \\ &= \frac{1}{2} \left(\bar{U}_{n,3} \left[\bar{S}_{n,1} + \bar{S}_{n,2} + \tau \bar{U}_{n,3}^\top M_x \mathcal{N}(\bar{U}_{n,3} \bar{S}_{n,2} \bar{V}_{n,3}^\top) M_y \bar{V}_{n,3} \right] \bar{V}_{n,3}^\top, \mathbf{I} \right)_M \\ &= \left(\bar{U}_{n,3} \bar{S}_{n,1} \bar{V}_{n,3}^\top, \mathbf{I} \right)_M + \frac{\tau}{2} \left(\mathcal{N}(\bar{U}_{n,3} \bar{S}_{n,1} \bar{V}_{n,3}^\top), \mathbf{I} \right)_M + \frac{\tau}{2} \left(\mathcal{N}(\bar{U}_{n,3} \bar{S}_{n,2} \bar{V}_{n,3}^\top), \mathbf{I} \right)_M \\ &= \left(\bar{U}_{n,3} \bar{U}_{n,3}^\top U_{n,1} S_{n,1} V_{n,1}^\top \bar{V}_{n,3} \bar{V}_{n,3}^\top, \mathbf{I} \right)_M = \left(U_{n,1} S_{n,1} V_{n,1}^\top, \mathbf{I} \right)_M, \end{aligned} \quad (3.36)$$

where we have used (2.23). The Truncation step in Algorithm 3 implies

$$\begin{aligned} &\left| (U_{n,3} S_{n,3} V_{n,3}^\top, \mathbf{I})_M - (\bar{U}_{n,3} \bar{S}_{n,3} \bar{V}_{n,3}^\top, \mathbf{I})_M \right| \\ &\leq \left\| U_{n,3} S_{n,3} V_{n,3}^\top - \bar{U}_{n,3} \bar{S}_{n,3} \bar{V}_{n,3}^\top \right\|_M \|\mathbf{I}\|_M \leq C\eta. \end{aligned} \quad (3.37)$$

Similar to (3.34), the second linear step implies

$$(U_{n+1} S_{n+1} V_{n+1}, \mathbf{I})_M = (U_{n,3} S_{n,3} V_{n,3}^\top, \mathbf{I})_M.$$

Therefore, it holds

$$\left| (U_{n+1} S_{n+1} V_{n+1}^\top, \mathbf{I})_M - (U_n S_n V_n^\top, \mathbf{I})_M \right| \leq C\eta.$$

which implies the conclusion. \square

Remark 3.9 The proposed method is modified to improve mass conservation. In particular, the inclusion of q_m and q_n in Algorithm 3 in the basis ensures that the augmented solution exactly conserves mass. After truncation, however, this property is no longer strictly satisfied; as implied by Theorem 3.1, the resulting mass error is proportional to the truncation tolerance at each time step. We also note that this error may accumulate over time. Addressing the removal of this error will be left for future work.

To give the discrete energy law for the low-rank solution, we introduce the following assumptions.

Assumption 3.1 (i) \mathcal{N} is Lipschitz-continuous and bounded [15]: for all $X, Y \in \mathbb{R}^{m \times n}$,

$$\|\mathcal{N}(X) - \mathcal{N}(Y)\|_{\mathbf{M}} \leq C_N \|X - Y\|_{\mathbf{M}}.$$

(ii) For coefficient matrix $\mathcal{W}(t) \in \mathcal{M}_{\bar{r}}$ to the low rank solution $\mathbf{w}_h \in \overline{\mathcal{Q}}_3^{n,k}$, it holds that projection error is more accurate than the tangent-space projection for approximating $\mathcal{N}(\mathcal{W})$ (see [40] Proposition.3)

$$\|\mathcal{N}(\mathcal{W}) - \overline{U}_{n,3} \overline{U}_{n,3}^{\top} M_x \mathcal{N}(\mathcal{W}) M_y \overline{V}_{n,3} \overline{V}_{n,3}^{\top}\|_{\mathbf{M}} \leq \|\mathcal{N}(\mathcal{W}) - \mathcal{P}_{\mathcal{W}}(\mathcal{N}(\mathcal{W}))\|_{\mathbf{M}} \leq \mu,$$

where the bases are given in Algorithm 3 and

$$\mu = \sup_{[t_n, t_{n+1}]} \sup_{\mathcal{W} \in \mathcal{M}_{\bar{r}}} \|\mathcal{N}(\mathcal{W}) - \mathcal{P}_{\mathcal{W}}(\mathcal{N}(\mathcal{W}))\|_{\mathbf{M}}.$$

Then we are ready to state the following modified energy law.

Theorem 3.2 Assume the Assumption 3.1 hold. For both the classical AC equation (1.1) and the conservative AC equation (1.2), if the time step τ satisfies the conditions stated in Lemma 2.4, the BUG-MLFEM solution $\mathbf{w}_h^{n+1} = \Phi(x)^{\top} U_{n+1} S_{n+1} V_{n+1}^{\top} \Psi(y)$, obtained via Algorithm 3 or (3.29), satisfies the following modified energy law

$$\tilde{E}^{n+1} \leq \tilde{E}^n - \beta^2 + \left(\frac{\eta}{\tau} + \frac{(1 + C_N \tau) \mu}{2} \right) \beta, \quad (3.38)$$

where $\beta = \|\mathbf{w}_h^{n+1,1} - \mathbf{w}_h^{n,1}\|_h$, and the modified energy

$$\tilde{E}^n = \tilde{E}(\mathbf{w}_h^{n,1}) = \frac{1}{2\tau} \left[(e^{-\tau \epsilon^2 \mathcal{L}_h} - 1) \mathbf{w}_h^{n,1}, \mathbf{w}_h^{n,1} \right]_h + (G(\mathbf{w}_h^{n,1}), 1)_h, \quad (3.39)$$

with G being defined in (2.33).

Proof From (3.25), it holds

$$\begin{cases} \overline{U}_{n,3} \overline{S}_{n,2} \overline{V}_{n,3}^{\top} = \overline{U}_{n,3} \overline{S}_{n,1} \overline{V}_{n,3}^{\top} + \tau \overline{U}_{n,3} \overline{U}_{n,3}^{\top} M_x \mathcal{N}(\overline{U}_{n,3} \overline{S}_{n,1} \overline{V}_{n,3}^{\top}) M_y \overline{V}_{n,3} \overline{V}_{n,3}^{\top} \\ \overline{U}_{n,3} \overline{S}_{n,3} \overline{V}_{n,3}^{\top} = \frac{1}{2} \overline{U}_{n,3} \overline{S}_{n,1} \overline{V}_{n,3}^{\top} + \frac{1}{2} \overline{U}_{n,3} \overline{S}_{n,2} \overline{V}_{n,3}^{\top} \\ \quad + \frac{\tau}{2} \overline{U}_{n,3} \overline{U}_{n,3}^{\top} M_x \mathcal{N}(\overline{U}_{n,3} \overline{S}_{n,2} \overline{V}_{n,3}^{\top}) M_y \overline{V}_{n,3} \overline{V}_{n,3}^{\top}. \end{cases} \quad (3.40)$$

Based on (3.33), we define

$$\overline{\mathcal{W}}_{n,3} = \overline{U}_{n,3} \overline{S}_{n,3} \overline{V}_{n,3}^{\top}. \quad (3.41)$$

and

$$\begin{aligned}
 \overline{\mathcal{W}}_{n,2} &= \overline{U}_{n,3} \overline{S}_{n,2} \overline{V}_{n,3}^\top \\
 &= \overline{U}_{n,3} \overline{S}_{n,1} \overline{V}_{n,3}^\top + \tau \overline{U}_{n,3} \overline{U}_{n,3}^\top M_x \mathcal{N}(\overline{U}_{n,3} \overline{S}_{n,1} \overline{V}_{n,3}^\top) M_y \overline{V}_{n,3} \overline{V}_{n,3}^\top \\
 &= U_{n,1} S_{n,1} V_{n,1}^\top + \tau \overline{U}_{n,3} \overline{U}_{n,3}^\top M_x \mathcal{N}(U_{n,1} S_{n,1} V_{n,1}^\top) M_y \overline{V}_{n,3} \overline{V}_{n,3}^\top \\
 &= \mathcal{W}_{n,1} + \tau \overline{U}_{n,3} \overline{U}_{n,3}^\top M_x \mathcal{N}(\mathcal{W}_{n,1}) M_y \overline{V}_{n,3} \overline{V}_{n,3}^\top.
 \end{aligned} \tag{3.42}$$

Thus, it can be concluded from (3.40), (3.41) and (3.42):

$$\overline{\mathcal{W}}_{n,3} - \mathcal{W}_{n,1} = \frac{\tau}{2} \overline{U}_p \mathcal{N}(\mathcal{W}_{n,1}) \overline{V}_p + \frac{\tau}{2} \overline{U}_p \mathcal{N}(\overline{\mathcal{W}}_{n,2}) \overline{V}_p, \tag{3.43}$$

where $\overline{U}_p = \overline{U}_{n,3} \overline{U}_{n,3}^\top M_x$ and $\overline{V}_p = \overline{V}_{n,3} \overline{V}_{n,3}^\top M_y$ denote the orthogonal projector on to the range of $\overline{U}_{n,3}$ and $\overline{V}_{n,3}$, respectively. (3.43) can be reformulated as

$$\frac{1}{\tau} (\overline{\mathcal{W}}_{n,3} - \mathcal{W}_{n+1,1}) + \frac{1}{\tau} (\mathcal{W}_{n+1,1} - \mathcal{W}_{n,1}) = \frac{1}{2} \overline{U}_p \mathcal{N}(\mathcal{W}_{n,1}) \overline{V}_p + \frac{1}{2} \overline{U}_p \mathcal{N}(\overline{\mathcal{W}}_{n,2}) \overline{V}_p. \tag{3.44}$$

The first term of the left side of (3.44), taking M-weighted inner product with $\mathcal{W}_{n+1,1} - \mathcal{W}_{n,1}$, gives

$$\begin{aligned}
 &\frac{1}{\tau} (\overline{\mathcal{W}}_{n,3} - \mathcal{W}_{n+1,1}, \mathcal{W}_{n+1,1} - \mathcal{W}_{n,1})_M \\
 &= \frac{1}{\tau} (\overline{\mathcal{W}}_{n,3} - \mathcal{W}_{n,3}, \mathcal{W}_{n+1,1} - \mathcal{W}_{n,1})_M + \frac{1}{\tau} (\mathcal{W}_{n,3} - \mathcal{W}_{n+1,1}, \mathcal{W}_{n+1,1} - \mathcal{W}_{n,1})_M.
 \end{aligned} \tag{3.45}$$

By (3.26), the first term on the right hand side of (3.45) implies

$$\begin{aligned}
 &\frac{1}{\tau} (\overline{\mathcal{W}}_{n,3} - \mathcal{W}_{n,3}, \mathcal{W}_{n+1,1} - \mathcal{W}_{n,1})_M \\
 &\leq \frac{1}{\tau} \|\overline{\mathcal{W}}_{n,3} - \mathcal{W}_{n,3}\|_M \|\mathcal{W}_{n+1,1} - \mathcal{W}_{n,1}\|_M \leq \frac{\eta}{\tau} \|\mathcal{W}_{n+1,1} - \mathcal{W}_{n,1}\|_M.
 \end{aligned}$$

For linear step in Algorithm 3,

$$\mathcal{W}_{n,3} = e^{-\tau \epsilon^2 \mathcal{L}} \mathcal{W}_{n+1,1}.$$

Then the second term on the right hand side of (3.45) holds

$$\begin{aligned}
 &\frac{1}{\tau} (\mathcal{W}_{n,3} - \mathcal{W}_{n+1,1}, \mathcal{W}_{n+1,1} - \mathcal{W}_{n,1})_M \\
 &= \frac{1}{\tau} (e^{-\tau \epsilon^2 \mathcal{L}} \mathcal{W}_{n+1,1} - \mathcal{W}_{n+1,1}, \mathcal{W}_{n+1,1} - \mathcal{W}_{n,1})_M \\
 &= \frac{1}{2\tau} \left[(e^{-\tau \epsilon^2 \mathcal{L}} \mathcal{W}_{n+1,1} - \mathcal{W}_{n+1,1}, \mathcal{W}_{n+1,1})_M - (e^{-\tau \epsilon^2 \mathcal{L}} \mathcal{W}_{n,1} - \mathcal{W}_{n,1}, \mathcal{W}_{n,1})_M \right]
 \end{aligned}$$

$$+ (e^{-\tau\epsilon^2\mathcal{L}}(\mathcal{W}_{n+1,1} - \mathcal{W}_{n,1}), \mathcal{W}_{n+1,1} - \mathcal{W}_{n,1})_M - \|\mathcal{W}_{n+1,1} - \mathcal{W}_{n,1}\|_M^2, \quad (3.46)$$

where we use the identity

$$(A - B, 2A)_M = \|A\|_M^2 - \|B\|_M^2 + \|A - B\|_M^2.$$

The right side of (3.44), taking M -weighted inner product with $\mathcal{W}_{n+1,1} - \mathcal{W}_{n,1}$, gives

$$\begin{aligned} & \frac{1}{2} (\bar{U}_p \mathcal{N}(\mathcal{W}_{n,1}) \bar{V}_p + \bar{U}_p \mathcal{N}(\bar{\mathcal{W}}_{n,2}) \bar{V}_p, \mathcal{W}_{n+1,1} - \mathcal{W}_{n,1})_M \\ &= \frac{1}{2} (\bar{U}_p \mathcal{N}(\mathcal{W}_{n,1}) \bar{V}_p + \bar{U}_p \mathcal{N}(\bar{\mathcal{W}}_{n,2}) \bar{V}_p \\ & \quad - (\mathcal{N}(\mathcal{W}_{n,1}) + \mathcal{N}(\bar{U}_{n,3} \bar{S}_{n,2} \bar{V}_{n,3}^\top), \mathcal{W}_{n+1,1} - \mathcal{W}_{n,1})_M \\ & \quad + \frac{1}{2} ((\mathcal{N}(\mathcal{W}_{n,1}) + \mathcal{N}(\bar{U}_{n,3} \bar{S}_{n,2} \bar{V}_{n,3}^\top), \mathcal{W}_{n+1,1} - \mathcal{W}_{n,1})_M \\ & \leq \frac{1}{2} (\mathcal{N}(\mathcal{W}_{n,1}) + \mathcal{N}(\bar{U}_{n,3} \bar{S}_{n,2} \bar{V}_{n,3}^\top), \mathcal{W}_{n+1,1} - \mathcal{W}_{n,1})_M + \frac{\mu}{2} \|\mathcal{W}_{n+1,1} - \mathcal{W}_{n,1}\|_M \\ & \leq \frac{1}{2} (\mathcal{N}(\mathcal{W}_{n,1}) + \mathcal{N}(\mathcal{W}_{n,1} + \tau \mathcal{N}(\mathcal{W}_{n,1})), \mathcal{W}_{n+1,1} - \mathcal{W}_{n,1})_M + \frac{\mu}{2} \|\mathcal{W}_{n+1,1} - \mathcal{W}_{n,1}\|_M \\ & \quad + \frac{1}{2} (\mathcal{N}(\bar{U}_{n,3} \bar{S}_{n,2} \bar{V}_{n,3}^\top) - \mathcal{N}(\mathcal{W}_{n,1} + \tau \mathcal{N}(\mathcal{W}_{n,1})), \mathcal{W}_{n+1,1} - \mathcal{W}_{n,1})_M \\ & \leq \frac{1}{2} (\mathcal{N}(\mathcal{W}_{n,1}) + \mathcal{N}(\mathcal{W}_{n,1} + \tau \mathcal{N}(\mathcal{W}_{n,1})), \mathcal{W}_{n+1,1} - \mathcal{W}_{n,1})_M + \frac{\mu}{2} \|\mathcal{W}_{n+1,1} - \mathcal{W}_{n,1}\|_M \\ & \quad + \frac{\tau C_N}{2} \|\bar{U}_{n,3} \bar{U}_{n,3}^\top M_x \mathcal{N}(\mathcal{W}_{n,1}) M_y \bar{V}_{n,3} \bar{V}_{n,3}^\top - \mathcal{N}(\mathcal{W}_{n,1})\|_M \|\mathcal{W}_{n+1,1} - \mathcal{W}_{n,1}\|_M \\ & \leq -(g(\mathcal{W}_{n,1}), \mathcal{W}_{n+1,1} - \mathcal{W}_{n,1})_M + \frac{\mu \tau C_N}{2} \|\mathcal{W}_{n+1,1} - \mathcal{W}_{n,1}\|_M + \frac{\mu}{2} \|\mathcal{W}_{n+1,1} - \mathcal{W}_{n,1}\|_M \\ & \leq -(g(\mathcal{W}_{n,1}), \mathcal{W}_{n+1,1} - \mathcal{W}_{n,1})_M + \frac{(1 + C_N \tau) \mu}{2} \|\mathcal{W}_{n+1,1} - \mathcal{W}_{n,1}\|_M, \end{aligned}$$

where g is defined in (2.34), and Assumption 3.1 has been applied in the first, third, and fourth inequalities.

Combining the results above, we obtain

$$\begin{aligned} & \tilde{E}(\mathbf{w}_h^{n+1,1}) - \tilde{E}(\mathbf{w}_h^{n,1}) \\ &= \frac{1}{2\tau} \left[(e^{-\tau\mathcal{L}_h} - 1) \mathbf{w}_h^{n+1,1}, \mathbf{w}_h^{n+1,1} \right]_h - (e^{-\tau\mathcal{L}_h} - 1) \mathbf{w}_h^{n,1}, \mathbf{w}_h^{n,1} \Big]_h \\ & \quad + (G(\mathbf{w}_h^{n+1,1}), 1)_h - (G(\mathbf{w}_h^{n,1}), 1)_h \Big] \\ & \leq - \left(\frac{1}{2\tau} - \frac{g'(\xi)}{2} \right) \|\mathbf{w}_h^{n+1,1} - \mathbf{w}_h^{n,1}\|_h^2 + \frac{\eta}{\tau} \|\mathbf{w}_h^{n+1,1} \\ & \quad - \mathbf{w}_h^{n,1}\|_h + \frac{(1 + C_N \tau) \mu}{2} \|\mathbf{w}_h^{n+1,1} - \mathbf{w}_h^{n,1}\|_h, \quad (3.47) \end{aligned}$$

which yields the energy dissipation law (3.38). \square

Remark 3.10 The energy dissipation law (3.38) for the BUG-MLFEM solution has an additional term, $(\frac{\eta}{\tau} + (1 + C\tau)\mu)\beta - \beta^2$, which accounts for the truncation error and projection error in Assumption 3.1. To ensure that Algorithm 3 achieves second-order accuracy in time, it is reasonable to assume

$$\eta = \mathcal{O}(\tau^2), \quad \text{and} \quad \beta \sim \|\mathbf{w}_h^{n+1} - \mathbf{w}_h^n\|_h = \mathcal{O}(\tau(\tau^2 + h^{k+1})).$$

Under these assumptions, (3.38) reduces to

$$\tilde{E}^{n+1} \leq \tilde{E}^n + \mathcal{O}(\tau(\tau^2 + h^{k+1})),$$

which implies that the last two terms in (3.38) are higher-order error terms. Consequently, the energy dissipation property is preserved up to an error significantly smaller than the overall approximation accuracy.

Remark 3.11 In algorithm implementation, the choice of the truncation threshold η is flexible. In this paper, we set $\eta = \alpha \|\Sigma\|_2$, with an approximate α to retain sufficient accuracy of $\bar{S}_{n,3}$. Numerical experiments demonstrate that this choice allows the low-rank solution to preserve the essential structure of the full-rank solution, including comparable energy dissipation behavior and mass conservation.

4 Numerical experiment

In this section, we present some numerical experiments to demonstrate the advantages of the proposed method.

4.1 Convergence test

Example 4.1 We first verify the spatial and temporal convergence of the proposed schemes on classical Allen-Cahn equation with a smooth initial condition

$$u(x, y, t = 0) = \sin(\pi x) \sin(\pi y), \quad (x, y) \in [0, 1] \times [0, 1]. \quad (4.1)$$

We set $\epsilon = 0.01$ and terminal time $T = 1.0$. Since the exact solution is unavailable, the temporal error is computed by fixing the spatial mesh size $h = 1/256$ and comparing the numerical solutions with a reference solution obtained using a very small time step ($\tau = 0.0001$). Similarly, the spatial error is computed by fixing $\tau = 0.0001$ and comparing the numerical solutions with a reference solution obtained using a refined mesh (with $h = 1/512$). The numerical errors are computed as:

$$L^2 \text{ error} = \|\mathbf{w}_{\text{ref}} - \mathbf{w}_h\|_h.$$

where \mathbf{w}_{ref} is the reference solution.

The spatial errors at the final time T , computed using the FR-MLFEM and BUG-MLFEM for the classical AC equation, are presented in Figure 1. As expected, the

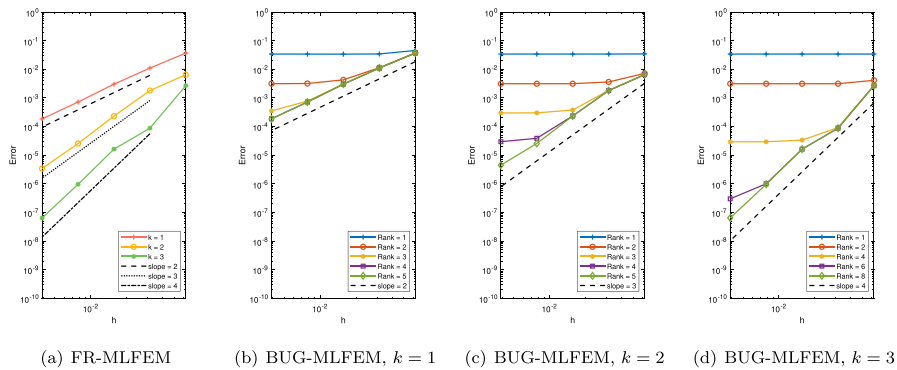


Fig. 1 Spatial accuracy tests of the BUG-MLFEM at $T = 1$

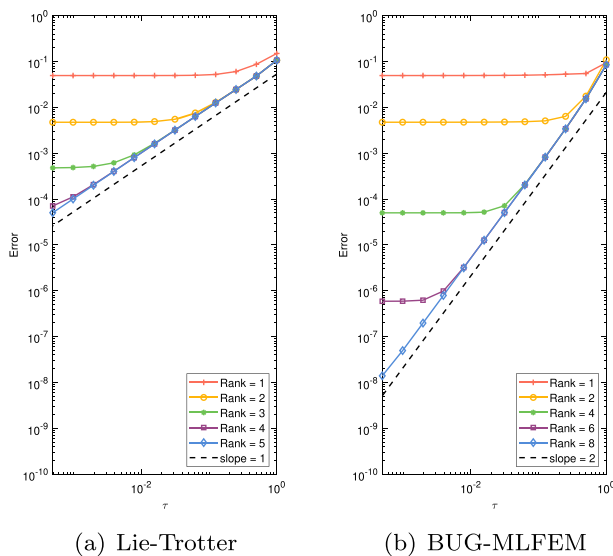


Fig. 2 Temporal accuracy tests of the BUG-MLFEM at $T = 1$

FR-MLFEM achieves $(k + 1)$ -th order accuracy in space (Figure 1(a)). Figure 1(b)–(d) report the errors and spatial convergence rates for the BUG-MLFEM with different rank r . We observe that when r is small, the truncation error dominates. However, for a moderately large rank ($r = 8$ for this example), the same errors and convergence rates as the full-rank solution are achieved.

The temporal errors are presented in Figure 2. In Figure 2(a), we additionally provide the errors and convergence rates computed by the first order Lie-Trotter integrator (see Remark 3.6).

The results demonstrate that the Lie-Trotter dynamical low-rank finite element solution achieves first-order accuracy in time, while the proposed Algorithm 3 attains second-order accuracy in time.

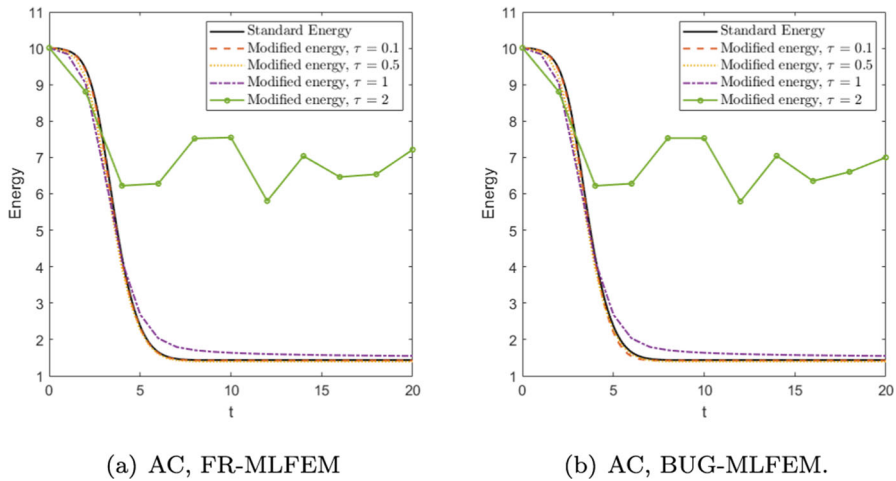


Fig. 3 Evolution of Energy and modified energy calculated by different methods; Left: FR-MLFEM. Middle: BUG-MLFEM. Right: Rank. Parameters: $N_x = N_y = 513$, tolerance $\eta = 0.01 \|\Sigma\|_2$

4.2 Modified energy

Example 4.2 In this example, we consider the classical AC equation with domain $\Omega = [0, 2\pi] \times [0, 2\pi]$ and the initial data choose as

$$u(x, y, t = 0) = 0.05 \sin x \sin y. \quad (4.2)$$

We solve this problem using the FR-MLFEM and the BUG-MLFEM, with parameters set to $\epsilon = 0.01$, $k = 1$, $N_x = N_y = 512$. The rank adaptive tolerance is chosen as $0.01 \|\Sigma\|_2$. In Figure 3, we plot the standard energy (computed by the FR-MLFEM with $\tau = 0.01$) and the modified energy of the numerical solutions. Figure 3(a) shows that as τ increases from 0.1 to 2, the modified energy curve gradually deviates from the standard energy curve, eventually becoming oscillatory at $\tau = 2$. However, when τ is small ($\tau = 0.1$), there is no significant difference between the modified energy and standard energy curves. The energy curves of the low-rank solutions are similar to those of the full-rank solution, consistent with the theoretical results in Lemma 2.4 for the full-rank scheme and Theorem 3.2 for the low-rank scheme. In Figure 3, it can also be seen that the energy becomes oscillatory when $\tau = 2$, which is consistent with the result in Lemma 2.4 and Theorem 3.2 stating that the modified energy is stable only for $\tau \leq 1/C_g$ for some constant C_g .

Moreover, in this example, we fix $\tau = 0.1$ and set the final time $T = 100$, and compare the CPU time of the FR-MLFEM and BUG-MLFEM while achieving comparable numerical results across varying parameters, as shown in Table 2. The evolution of the rank is illustrated in Figure 4. The results indicate that the BUG-MLFEM is substantially faster than the FR-MLFEM as expected, owing to its lower computational complexity.

Table 2 CPU time of the numerical methods.

Method	h	k	m(=n)	CPU/s	Rate
FR-MLFEM	$2\pi/512$	1	513	10.831	
	$2\pi/512$	2	1025	53.597	2.3070
	$2\pi/1024$	2	2049	285.765	2.4146
	$2\pi/2048$	2	4096	1891.932	2.7270
BUG-MLFEM ($\eta = 0.001 \ \Sigma\ _2$)	$2\pi/512$	1	513	2.8465	
	$2\pi/512$	2	1025	4.0354	0.9676
	$2\pi/1024$	2	2049	7.3826	1.2459
	$2\pi/2048$	2	4096	80.423	1.9347

4.3 Mass-conservation

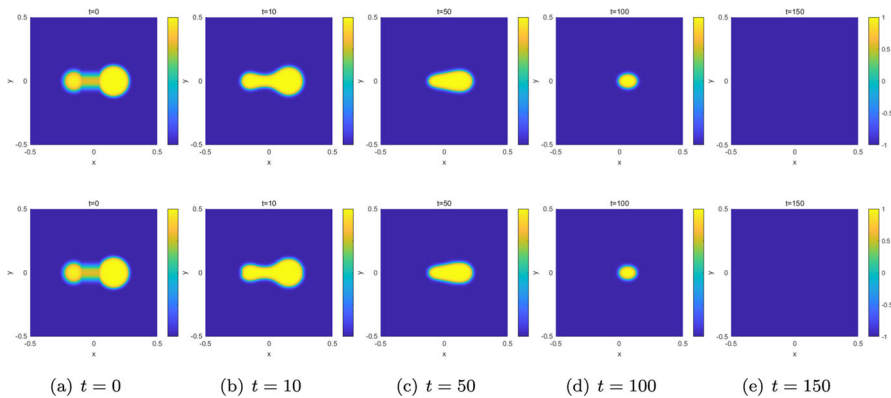
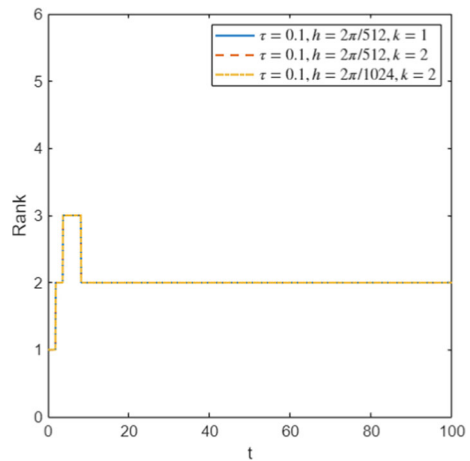
Example 4.3 Consider the classical AC equation (1.1) and the conservative AC equation (1.2). The initial function is taken as an unequal barbell in $\Omega \in [-0.5, 0.5] \times [-0.5, 0.5]$,

$$u_0 = \begin{cases} \tanh\left(\frac{3}{\epsilon}\left((x - \frac{\alpha}{4})^2 + y^2 - (0.195\alpha)^2\right)\right), & x > 0.07, \\ \tanh\left(\frac{3}{\epsilon}\left(y^2 - (0.075\alpha)^2\right)\right), & -0.15 \leq x \leq 0.07, \\ \tanh\left(\frac{3}{\epsilon}\left((x + \frac{\alpha}{4})^2 + y^2 - (0.195\alpha)^2\right)\right), & x < -0.15, \end{cases} \quad (4.3)$$

The parameters are chosen as $\epsilon = 0.01$, $\alpha = \frac{5}{8}$. In addition, the parameters for both FR-MLFEM and BUG-MLFEM are chosen as $k = 1$, $N_x = N_y = 512$, $\tau = 0.1$.

We first solve the classical AC equation. The snapshots of the solutions computed by FR-MLFEM and BUG-MLFEM at different times are presented in Figure 5. As time evolves, the small bell begin to be absorbed by the bigger bell, coalescing into a single shrinking bubble, which eventually disappears. There is no visible difference between the full-rank and low-rank solutions. The evolutions of the mass changes, original energy, and rank are presented in Figure 7. From these results, we observe that: (a) the solution of the classical AC equation does not conserve mass, (b) both the full-rank and low-rank solutions satisfy the energy dissipation law, and (c) the rank in the low-rank algorithm decreases as the solution evolves toward the steady state.

Next, we solve the conservative AC equation with RSLM using the same initial condition. The snapshots of the solutions computed by FR-MLFEM and BUG-MLFEM are presented in Figure 6. In this case, the two bells coalesce into a larger, increasingly round bubble. Compared to the classical Allen-Cahn equation, the bubble does not disappear. From Figure 7, we can observe that (a) both the full-rank and low-rank solutions conserve mass very well, (b) both the full-rank and low-rank solutions satisfy the energy dissipation law, and (c) the rank in the low-rank algorithm is no more

Fig. 4 Rank**Fig. 5** Snapshots of solutions of AC equation computed using FR-MLFEM (first row) and BUG-MLFEM (second row). Parameters: $N_x = N_y = 513$, $\tau = 0.1$, $\eta = 0.01 \|\Sigma\|_2$

than $r = 9$. The result of AC equation with BBLM is similarly to the RSLM, to save place, we omit here.

4.4 Symmetry breaking

Symmetry-breaking phenomena are commonly observed in phase field simulations. Theoretically, for a given initial odd function in a square domain centered at the origin, the exact solution should always maintain odd symmetry. In applications and numerical simulations, numerical schemes are expected to preserve this symmetry over time. However, in practice, this symmetry is often broken in long simulations [35]. Among existing algorithms, IMEX [11], operator splitting methods [51], BDF2 [50], and ETDRK [15, 46] fail to preserve this symmetry, including the recently popular SAV approach [44, 45].

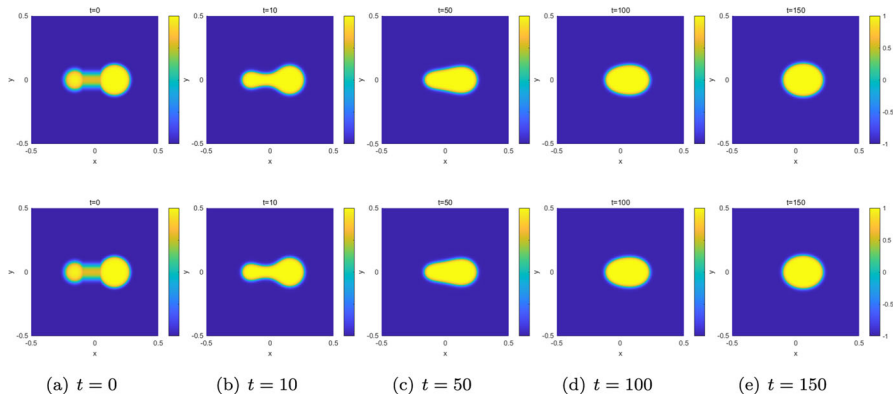


Fig. 6 Snapshots of solutions of AC equation with RSLM, computed using FR-MLFEM(first row) and BUG-MLFEM(second row). Parameters: $N_x = N_y = 513$, $\tau = 0.1$, $\eta = 0.001 \|\Sigma\|_2$.

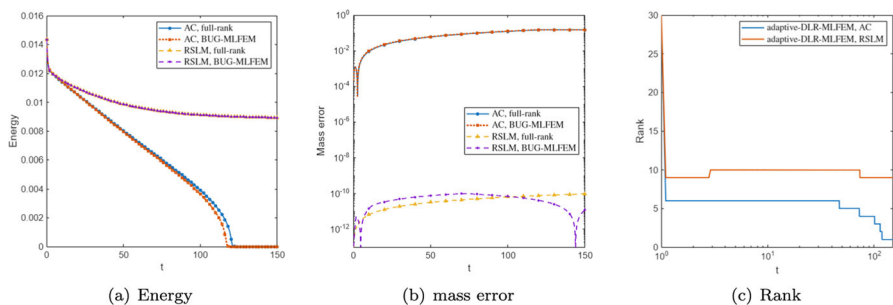


Fig. 7 Evolution of numerical solution properties. Left: energy; middle: mass error; right: Rank

Example 4.4 We consider the classical Allen-Cahn equation in domain $\Omega \in [-0.5, 0.5] \times [-0.5, 0.5]$ with different initial values

$$\begin{aligned} u_1 &= \sin(2\pi x)\sin(2\pi y), \\ u_2 &= \sin(2\pi x)\sin(4\pi y). \end{aligned} \quad (4.4)$$

We also consider a shifted initial value

$$u_3 = \sin\left(2\pi\left(x + \frac{\pi}{8}\right)\right)\sin\left(4\pi\left(y + \frac{\pi}{8}\right)\right). \quad (4.5)$$

The snapshots of solutions for different initial values are shown in Figure 8. Since the initial values in (4.4) and (4.5) are odd and smooth, the solutions should preserve odd symmetry over time, i.e., $u(-x, y, t) = -u(x, y, t)$ and $u(x, -y, t) = -u(x, y, t)$. Figure 9 and Figure 10 compare the numerical solutions of FR-MLFEM and BUG-MLFEM. From the results, it is observed that the odd symmetry of the solution computed by FR-MLFEM is broken, whereas the BUG-MLFEM preserves symmetry throughout the simulation.

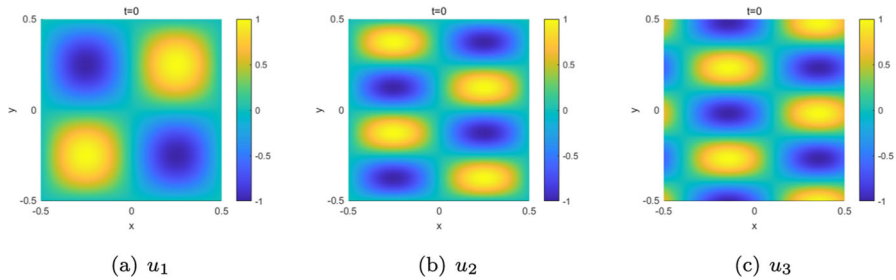


Fig. 8 Snapshots of different initial values

A similar numerical result for the case with the initial value u_1 was presented in [35], where symmetry-breaking, similar to that observed with the FR-MLFEM, was reported. To address this issue, a symmetry-preserving filter was introduced in [35] to maintain the odd symmetry of the solution. Notably, the proposed BUG-MLFEM method achieves consistent results by inherently preserving the symmetry without requiring any such filter.

For the case with the shifted initial value u_3 , snapshots of the full-rank and low-rank solutions are presented in Figure 11, where it is evident that the FR-MLFEM fails to preserve the symmetry at $t \approx 100$. The symmetry-preserving filter introduced in [35] may not be applicable to improve it. In contrast, the BUG-MLFEM method successfully maintains the symmetry structure. Figure 12 presents the evolution of energy, mass, and rank. We observe that the energy is non-increasing, and the rank remains 1 in all cases. Since this is the classical AC equation, which does not preserve mass, the masses of the full-rank solutions vary over time. In contrast, the low-rank solutions appear to conserve mass, in a certain sense, due to their symmetry-preserving behavior near steady state.

Remark 4.1 Symmetry breaking may arise from various sources, such as machine round-off errors, non-symmetric domain partitioning, or errors in numerical integration. Although these errors are typically small at each time step, they can accumulate during long-time simulations. When manifested in the coefficient matrices, such errors correspond to very small singular values. The truncation mechanism in BUG-MLFEM automatically removes these small singular values, thereby preventing their accumulation and preserving the pattern symmetry.

5 Concluding remarks

A second-order BUG-MLFEM was proposed to solve both the classical and conservative AC equations. The matrix differential equation arising from the semi-discrete mass-lumped finite element method is decomposed into linear and nonlinear components. The linear component is solved analytically, while the nonlinear component is addressed using the second-order augmented BUG integrator. The computational complexity of the proposed algorithm is $\mathcal{O}((m^2 + n^2)r)$. The mass is conserved up

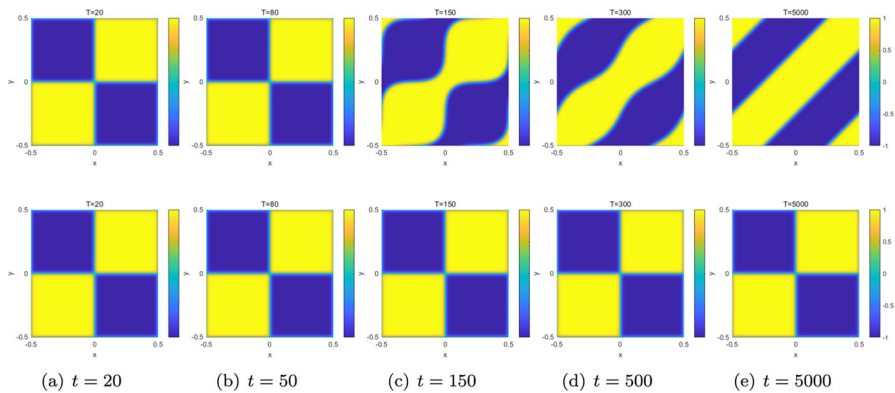


Fig. 9 Snapshots of solutions to the AC equation at different times with initial value u_1 , computed using the FR-MLFEM (first row) and the BUG-MLFEM (second row)

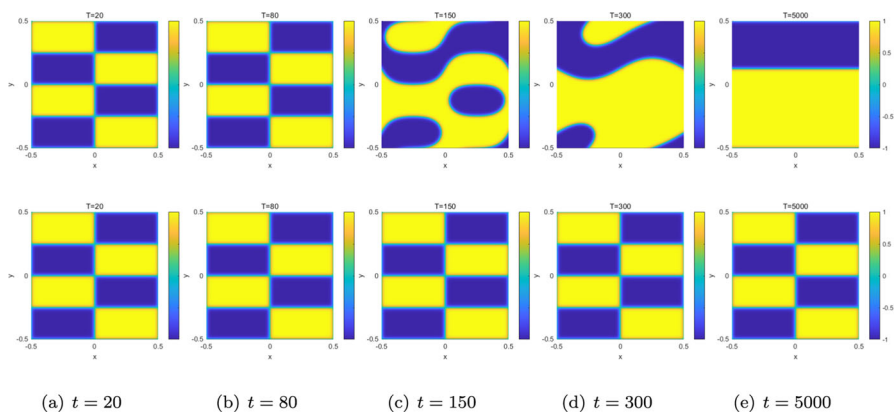


Fig. 10 Snapshots of solutions to the AC equation at different times with initial value u_2 , computed using the FR-MLFEM (first row) and the BUG-MLFEM (second row)

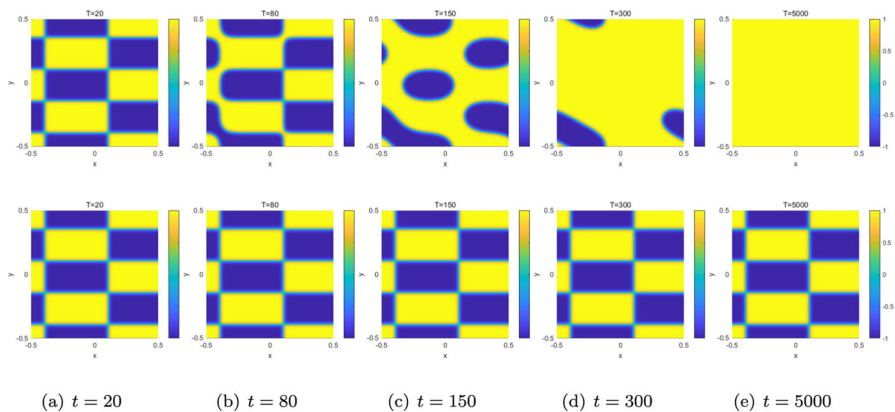


Fig. 11 Snapshots of solutions to the AC equation at different times with initial value u_3 , computed using the FR-MLFEM (first row) and the BUG-MLFEM (second row)

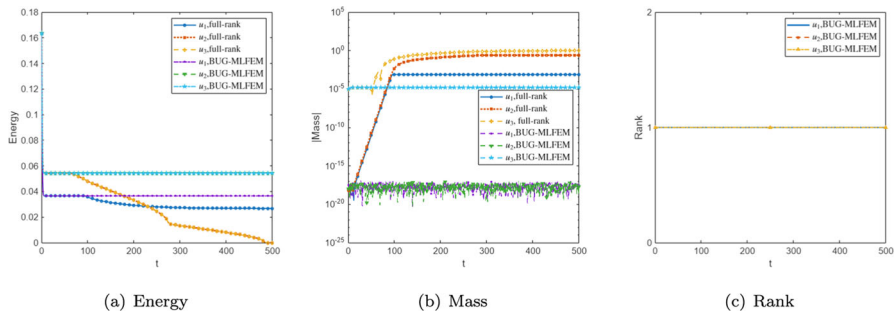


Fig. 12 Evolution of numerical solution properties. Left: Energy; middle: Mass; right: Rank

to the truncation tolerance in the conservative Allen-Cahn equation. The modified energy is dissipative up to a high-order error of $\mathcal{O}(\tau(\tau^2 + h^{k+1}))$. Hence, the energy stability property remains valid. Although some conservative BUG integrators have been developed in the literature (see, e.g., [19]), implementing them for the proposed method in the nonlinear problems is challenging, which will be addressed in future work. Numerical experiments demonstrate that the BUG-MLFEM solution preserves mass for the conservative AC equation and ensures energy dissipation. Additionally, the tests reveal that the BUG-MLFEM solution maintains symmetrical phenomena, highlighting its robustness for long-time simulations. Extending these results to three dimensions and exploring more general nonlinear terms, such as the logarithmic Flory-Huggins potential, would also be interesting directions for future research.

Appendix A Proofs and algorithms

In this section, we present some necessary proofs, and provide some useful algorithms.

A1 Proof of Lemma 2.2

Proof Let $q_m = [1, \dots, 1]^\top \in \mathbb{R}^m$ be a vector of ones. It can be verified that $(0, q_m)$ is an eigenpair of the matrix $L_x = -M_x^{-1}A_x$, i.e., $L_x q_m = 0q_m$. Then, it follows

$$e^{\tau \epsilon^2 L_x} q_m = \sum_{l=0}^{\infty} \frac{\tau^l}{l!} (\epsilon^2 L_x)^l q_m = \sum_{l=0}^{\infty} \frac{(0\tau)^l}{l!} q_m = I_m q_m = q_m,$$

which implies that $(0, q_m)$ is an eigenpair of the matrix $e^{\tau \epsilon^2 L_x}$. Similarly, $(0, q_n)$ is an eigenpair of the matrix $e^{\tau \epsilon^2 L_y}$, i.e., $e^{\tau \epsilon^2 L_y} q_n = q_n$, where q_n is also a vector of ones.

Next, we state that the matrices $M_x e^{\tau \epsilon^2 L_x}$ and $M_y e^{\tau \epsilon^2 L_y}$ are symmetric. We will prove this for $M_x e^{\tau \epsilon^2 L_x}$; the case for $M_y e^{\tau \epsilon^2 L_y}$ can be shown similarly.

$$\begin{aligned}
\left(M_x e^{\tau \epsilon^2 L_x}\right)^{\top} &= \left(e^{-\epsilon^2 \tau M_x^{-1} A_x}\right)^{\top} M_x^{\top} = e^{(-\epsilon^2 \tau M_x^{-1} A_x)^{\top}} M_x^{\top} \\
&= e^{-\epsilon^2 \tau A_x^{\top} (M_x^{-1})^{\top}} M_x^{\top} = e^{-\epsilon^2 \tau A_x M_x^{-1}} M_x^{\top} \\
&= M_x M_x^{-1} e^{-\epsilon^2 \tau A_x M_x^{-1}} M_x = M_x e^{-\epsilon^2 \tau M_x^{-1} A_x M_x^{-1} M_x} = M_x e^{\tau \epsilon^2 L_x}.
\end{aligned}$$

Then, using the cyclic property of the Frobenius inner product gives

$$\begin{aligned}
(e^{\tau \epsilon^2 \mathcal{L}} Z, \mathbf{I})_{\mathbf{M}} &= (M_x e^{\tau \epsilon^2 L_x} Z e^{\tau \epsilon^2 L_y^{\top}} M_y, \mathbf{I})_{\mathbf{F}} = (Z, (M_x e^{\tau \epsilon^2 L_x})^{\top} \mathbf{I} (e^{\tau \epsilon^2 L_y^{\top}} M_y)^{\top})_{\mathbf{F}} \\
&= (Z, M_x e^{\tau \epsilon^2 L_x} q_m q_n^{\top} M_y e^{\tau \epsilon^2 L_y})_{\mathbf{F}} \\
&= (Z, M_x e^{\tau \epsilon^2 L_x} q_m ((M_y e^{\tau \epsilon^2 L_y})^{\top} q_n)^{\top})_{\mathbf{F}} \\
&= (Z, M_x e^{\tau \epsilon^2 L_x} q_m (M_y e^{\tau \epsilon^2 L_y} q_n)^{\top})_{\mathbf{F}} \\
&= (Z, M_x q_m (M_y q_n)^{\top})_{\mathbf{F}} = (M_x Z M_y, \mathbf{I})_{\mathbf{F}} = (Z, \mathbf{I})_{\mathbf{M}}.
\end{aligned}$$

□

A2 Proof of Lemma 2.4

Proof Recall that $W_{n+1,1} = e^{\frac{\tau}{2} \epsilon^2 \mathcal{L}} W_{n+1} = S_{\frac{\tau}{2}}^{\mathcal{L}} W_{n+1}$. Using (2.31), it can be written as

$$W_{n+1,1} = S_{\frac{\tau}{2}}^{\mathcal{L}} S_{\frac{\tau}{2}}^{\mathcal{L}} S_{\tau}^{\mathcal{N}} W_{n,1} = S_{\tau}^{\mathcal{L}} S_{\tau}^{\mathcal{N}} W_{n,1},$$

which is equivalent to

$$S_{-\tau}^{\mathcal{L}} W_{n+1,1} = S_{\tau}^{\mathcal{N}} W_{n,1}.$$

Then it holds

$$\frac{1}{\tau} \left(S_{-\tau}^{\mathcal{L}} W_{n+1,1} - W_{n+1,1} \right) + \frac{1}{\tau} (W_{n+1,1} - W_{n,1}) = \frac{1}{\tau} (S_{\tau}^{\mathcal{N}} W_{n,1} - W_{n,1}). \quad (\text{A.1})$$

By (2.31), it can show

$$S_{\tau}^{\mathcal{N}} W_{n,1} - W_{n,1} = \frac{\tau}{2} (\mathcal{N}(W_{n,1}) + \mathcal{N}(W_{n,2})).$$

Thus,

$$\frac{1}{\tau} \left(S_{-\tau}^{\mathcal{L}} W_{n+1,1} - W_{n+1,1} \right) + \frac{1}{\tau} (W_{n+1,1} - W_{n,1}) = \frac{1}{2} (\mathcal{N}(W_{n,1}) + \mathcal{N}(W_{n,2})). \quad (\text{A.2})$$

The equation (A.2) can be reformulated in finite element form as

$$\frac{1}{\tau} \left(e^{-\tau \epsilon^2 \mathcal{L}_h} w_h^{n+1,1} - w_h^{n+1,1} \right) + \frac{1}{\tau} (w_h^{n+1,1} - w_h^{n,1}) = \frac{1}{2} (\mathcal{N}(w_h^{n,1}) + \mathcal{N}(w_h^{n,2})).$$

By taking the inner product with $w_h^{n+1,1} - w_h^{n,1}$ and using the identity

$$(a - b, 2a)_h = \|a\|_h^2 - \|b\|_h^2 + \|a - b\|_h^2,$$

it follows

$$\begin{aligned} & \frac{1}{2\tau} \left[\left((e^{-\tau\epsilon^2\mathcal{L}_h} - 1)w_h^{n+1,1}, w_h^{n+1,1} \right)_h - \left((e^{-\tau\epsilon^2\mathcal{L}_h} - 1)w_h^{n,1}, w_h^{n,1} \right)_h \right. \\ & \quad + \left(e^{-\tau\epsilon^2\mathcal{L}_h}(w_h^{n+1,1} - w_h^{n,1}), w_h^{n+1,1} - w_h^{n,1} \right)_h - \left\| w_h^{n+1,1} - w_h^{n,1} \right\|_h^2 \Big] \\ & \quad + \frac{1}{\tau} \left\| w_h^{n+1,1} - w_h^{n,1} \right\|_h^2 + \left(g(w_h^{n,1}), w_h^{n+1,1} - w_h^{n,1} \right)_h = 0. \end{aligned} \quad (\text{A.3})$$

Expanding $G(w_h^{n+1,1})$ in Taylor series yields

$$\begin{aligned} & G(w_h^{n+1,1}) - G(w_h^{n,1}, 1)_h \\ & = (g(w_h^{n,1}), w_h^{n+1,1} - w_h^{n,1})_h + \frac{1}{2}(g'(\xi), (w_h^{n+1,1} - w_h^{n,1})^2)_h. \end{aligned} \quad (\text{A.4})$$

where ξ is a function between $w_h^{n+1,1}$ and $w_h^{n,1}$. Given $|g'(\xi)| \leq C_g$ and $\tau \leq \frac{1}{C_g}$, it holds

$$\tilde{E}(w_h^{n+1,1}) - \tilde{E}(w_h^{n,1}) \leq -\left(\frac{1}{2\tau} - \frac{1}{2}g'(\xi)\right) \|w_h^{n+1,1} - w_h^{n,1}\|_h^2 \leq 0.$$

□

A3 Reduced generalized QR decomposition

We introduce a reduced generalized QR (RGQR) decomposition to generate new bases $U \in \mathbb{R}^{m \times \hat{r}}$ and $V \in \mathbb{R}^{n \times \hat{r}}$ from matrices $K \in \mathbb{R}^{m \times r}$ and $L \in \mathbb{R}^{n \times r}$, respectively,

$$[U, V] = \text{RGQR}(K, L, M_x, M_y), \quad (\text{A.5})$$

where $U^\top M_x U = I_{\hat{r}}$, and $V^\top M_y V = I_{\hat{r}}$. This method can reduce the computational cost while maintaining the accuracy.

Algorithm 4 RGQR decomposition

Input: Matrices $K \in \mathbb{R}^{m \times r}$, $L \in \mathbb{R}^{n \times r}$, $M_x \in \mathbb{R}^{m \times m}$, and $M_y \in \mathbb{R}^{n \times n}$.

Output: Bases $U \in \mathbb{R}^{m \times \hat{r}}$, $V \in \mathbb{R}^{n \times \hat{r}}$ with $\hat{r} \leq r$.

- **Step 1:** Perform generalized QR (GQR) decompositions [55] and obtain:

$$[U_1, R_1] = \text{GQR}(K, M_x), \quad [V_1, P_1] = \text{GQR}(L, M_y),$$

where $U_1^\top M_x U_1 = I_r$, and $V_1^\top M_y V_1 = I_r$.

- **Step 2:** Perform singular value decomposition (SVD) on R_1 and P_1 :

$$[Q_u, \Sigma_u, \sim] = \text{svd}(R_1), \quad [Q_v, \Sigma_v, \sim] = \text{svd}(P_1).$$

Determine the numerical ranks r_u and r_v by identifying the number of singular values greater than 10^{-12} in Σ_u and Σ_v , respectively, and set $\hat{r} = \max(r_u, r_v)$.

- **Step 3:** Construct the reduced bases:

$$U = U_1 Q_u(:, 1:\hat{r}), \quad V = V_1 Q_v(:, 1:\hat{r}).$$

Acknowledgements Yi's research was partially supported by NSFC Project (12431014). Yin's research was supported by the University of Texas at El Paso Startup Award.

Data Availability Enquiries about data availability should be directed to the authors.

Declarations

Conflicts of Interest The authors declare that they have no conflict of interest.

References

1. Allen, S.M., Cahn, J.W.: A microscopic theory for antiphase boundary motion and its application to antiphase domain coarsening. *Acta Metall.* **27**(6), 1085–1095 (1979)
2. Bergamaschi, L., Vianello, M.: Efficient computation of the exponential operator for large, sparse, symmetric matrices. *Numerical linear algebra with appl.* **7**(1), 27–45 (2000)
3. Blanes, S., Kopylov, N., Seydaoglu, M.: Efficient scaling and squaring method for the matrix exponential. *SIAM J. Matrix Anal. Appl.* **46**(1), 74–93 (2025)
4. Brassel, M., Bretin, E.: A modified phase field approximation for mean curvature flow with conservation of the volume. *Math. Methods Appl. Sci.* **10**(34), 1157–1180 (2011)
5. Ceruti, G., Einkemmer, L., Kusch, J., Lubich, C.: A robust second-order low-rank BUG integrator based on the midpoint rule. *BIT Numer. Math.* **64**(3), 30 (2024)
6. Ceruti, G., Kusch, J., Lubich, C.: A rank-adaptive robust integrator for dynamical low-rank approximation. *BIT Numer. Math.* **62**(4), 1149–1174 (2022)
7. Ceruti, G., Kusch, J., Lubich, C.: A parallel rank-adaptive integrator for dynamical low-rank approximation. *SIAM J. Sci. Comput.* **46**(3), B205–B228 (2024)
8. Ceruti, G., Lubich, C.: An unconventional robust integrator for dynamical low-rank approximation. *BIT Numer. Math.* **62**(1), 23–44 (2022)
9. Chen, C.M., Thomée, V.: The lumped mass finite element method for a parabolic problem. *The ANZIAM J.* **26**(3), 329–354 (1985)
10. Chen, Y., Liu, H., Yi, N., Yin, P.: Unconditionally energy stable IEQ-FEMs for the Cahn-Hilliard equation and Allen-Cahn equation. *Numerical Algorithms*, pages 1–42, (2024)
11. Cheng, Y., Kurganov, A., Qu, Z., Tang, T.: Fast and stable explicit operator splitting methods for phase-field models. *J. Comput. Phys.* **303**, 45–65 (2015)
12. Cox, S.M., Matthews, P.C.: Exponential time differencing for stiff systems. *J. Comput. Phys.* **176**(2), 430–455 (2002)
13. Dirac, P.A.: Note on exchange phenomena in the Thomas atom. *Math. proc. Camb. philos. soc.* **26**, 376–385 (1930). (Cambridge University Press)
14. Du, Q., Ju, L., Li, X., Qiao, Z.: Maximum principle preserving exponential time differencing schemes for the nonlocal Allen-Cahn equation. *SIAM J. Numer. Anal.* **57**(2), 875–898 (2019)
15. Du, Q., Ju, L., Li, X., Qiao, Z.: Maximum bound principles for a class of semilinear parabolic equations and exponential time-differencing schemes. *SIAM Rev.* **63**(2), 317–359 (2021)
16. Einkemmer, L.: A low-rank algorithm for weakly compressible flow. *SIAM J. Sci. Comput.* **41**(5), A2795–A2814 (2019)
17. Einkemmer, L., Hu, J., Ying, L.: An efficient dynamical low-rank algorithm for the Boltzmann-BGK equation close to the compressible viscous flow regime. *SIAM J. Sci. Comput.* **43**(5), B1057–B1080 (2021)

18. Einkemmer, L., Kormann, K., Kusch, J., McClarren, R.G., Qiu, J.-M.: A review of low-rank methods for time-dependent kinetic simulations. *Journal of Computational Physics*, page 114191, (2025)
19. Einkemmer, L., Kusch, J., Schotthöfer, S.: Conservation properties of the augmented basis update & Galerkin integrator for kinetic problems. *arXiv preprint [arXiv:2311.06399](https://arxiv.org/abs/2311.06399)*, (2023)
20. Einkemmer, L., Lubich, C.: A low-rank projector-splitting integrator for the vlasov-Poisson equation. *SIAM J. Sci. Comput.* **40**(5), B1330–B1360 (2018)
21. Frenkel, J.: *Wave mechanics*, clarendon, (1934)
22. Gong, Y., Zhao, J., Wang, Q.: Arbitrarily high-order unconditionally energy stable SAV schemes for gradient flow models. *Comput. Phys. Commun.* **249**, 107033 (2020)
23. Grasedyck, L., Kressner, D., Tobler, C.: A literature survey of low-rank tensor approximation techniques. *GAMM-Mitteilungen* **36**(1), 53–78 (2013)
24. Hauck, C.D., Schnake, S.: A predictor-corrector strategy for adaptivity in dynamical low-rank approximations. *SIAM J. Matrix Anal. Appl.* **44**(3), 971–1005 (2023)
25. Higham, N.J.: The scaling and squaring method for the matrix exponential revisited. *SIAM Rev.* **51**(4), 747–764 (2009)
26. Hochbruck, M., Lubich, C.: On krylov subspace approximations to the matrix exponential operator. *SIAM J. Numer. Anal.* **34**(5), 1911–1925 (1997)
27. Isherwood, L., Grant, Z.J., Gottlieb, S.: Strong stability preserving integrating factor Runge-Kutta methods. *SIAM J. Numer. Anal.* **56**(6), 3276–3307 (2018)
28. Ju, L., Li, X., Qiao, Z., Yang, J.: Maximum bound principle preserving integrating factor Runge-Kutta methods for semilinear parabolic equations. *J. Comput. Phys.* **439**, 110405 (2021)
29. Kim, J., Lee, S., Choi, Y.: A conservative allen-cahn equation with a space-time dependent lagrange multiplier. *Int. J. Eng. Sci.* **84**, 11–17 (2014)
30. Koch, O., Lubich, C.: Dynamical low-rank approximation. *SIAM J. Matrix Anal. Appl.* **29**(2), 434–454 (2007)
31. Kusch, J.: Second-order robust parallel integrators for dynamical low-rank approximation. *BIT Numer. Math.* **65**(3), 31 (2025)
32. Kusch, J., Ceruti, G., Einkemmer, L., Frank, M.: Dynamical low-rank approximation for burgers' equation with uncertainty. *Int. J. Uncertain. Quantif.* **62**(1), 23–44 (2022)
33. Lee, H.G., Shin, J., Lee, J.-Y.: A high-order and unconditionally energy stable scheme for the conservative allen-cahn equation with a nonlocal Lagrange multiplier. *J. Sci. Comput.* **90**, 1–12 (2022)
34. Li, B., Yang, J., Zhou, Z.: Arbitrarily high-order exponential cut-off methods for preserving maximum principle of parabolic equations. *SIAM J. Sci. Comput.* **42**(6), A3957–A3978 (2020)
35. Li, D., ChaoyuTang, T., Yang, W.: On symmetry breaking of allen-cahn. *CSIAM Trans. Appl. Math.* **3**(2), 221–243 (2022)
36. Li, D., Quan, C., Xu, J.: Stability and convergence of strang splitting. Part I: scalar allen-cahn equation. *J. Comput. Phys.* **458**, 111087 (2022)
37. Li, D., Quan, C., Xu, J.: Stability and convergence of strang splitting. Part II: tensorial allen-cahn equations. *J. Comput. Phys.* **454**, 110985 (2022)
38. Lubich, C., Oseledets, I.V.: A projector-splitting integrator for dynamical low-rank approximation. *BIT Numer. Math.* **54**(1), 171–188 (2014)
39. Nakao, J., Qiu, J.-M., Einkemmer, L.: Reduced augmentation Implicit low-rank (RAIL) integrators for advection-diffusion and Fokker-Planck models. *SIAM J. Sci. Comput.* **47**(2), A1145–A1169 (2025)
40. Nobile, F., Riffaud, S.: Robust high-order low-rank bug integrators based on explicit runge-kutta methods. *arXiv preprint [arXiv:2502.07040](https://arxiv.org/abs/2502.07040)*, (2025)
41. Peng, Z., McClarren, R.G., Frank, M.: A low-rank method for two-dimensional time-dependent radiation transport calculations. *J. Comput. Phys.* **421**, 109735 (2020)
42. Rubinstein, J., Sternberg, P.: Nonlocal reaction—diffusion equations and nucleation. *IMA J. Appl. Math.* **48**(3), 249–264 (1992)
43. Schotthöfer, S., Zangrando, E., Kusch, J., Ceruti, G., Tudisco, F.: Low-rank lottery tickets: finding efficient low-rank neural networks via matrix differential equations. *Adv. Neural. Inf. Process. Syst.* **35**, 20051–20063 (2022)
44. Shen, J., Xu, J., Yang, J.: The scalar auxiliary variable (SAV) approach for gradient flows. *J. Comput. Phys.* **353**, 407–416 (2018)
45. Shen, J., Xu, J., Yang, J.: A new class of efficient and robust energy stable schemes for gradient flows. *SIAM Rev.* **61**(3), 474–506 (2019)

46. Wang, X., Ju, L., Du, Q.: Efficient and stable exponential time differencing Runge-Kutta methods for phase field elastic bending energy models. *J. Comput. Phys.* **316**, 21–38 (2016)
47. Ward, M.J.: Metastable bubble solutions for the allen-cahn equation with mass conservation. *SIAM J. Appl. Math.* **56**(5), 1247–1279 (1996)
48. Weng, Z., Zhai, S., Dai, W., Yang, Y., Mo, Y.: Stability and error estimates of strang splitting method for the nonlocal ternary conservative allen-cahn model. *J. Comput. Appl. Math.* **441**, 115668 (2024)
49. Xiao, X., Gui, D., Feng, X.: A highly efficient operator-splitting finite element method for 2D/3D nonlinear allen-cahn equation. *International Journal of Numerical Methods for Heat & Fluid Flow* **27**(2), 530–542 (2017)
50. Xu, C., Tang, T.: Stability analysis of large time-stepping methods for epitaxial growth models. *SIAM J. Numer. Anal.* **44**(4), 1759–1779 (2006)
51. Xu, J., Li, Y., Wu, S., Bousquet, A.: On the stability and accuracy of partially and fully implicit schemes for phase field modeling. *Comput. Methods Appl. Mech. Eng.* **345**, 826–853 (2019)
52. Yang, J., Yi, N., Zhang, H.: High-order, unconditionally maximum-principle preserving finite element method for the allen-cahn equation. *Appl. Numer. Math.* **188**, 42–61 (2023)
53. Yang, X.: Linear, first and second-order, unconditionally energy stable numerical schemes for the phase field model of homopolymer blends. *J. Comput. Phys.* **327**, 294–316 (2016)
54. Yang, X., Ju, L.: Efficient linear schemes with unconditional energy stability for the phase field elastic bending energy model. *Comput. Methods Appl. Mech. Eng.* **315**, 691–712 (2017)
55. Yin, P., Endeve, E., Hauck, C., Schnake, S.: Towards dynamical low-rank approximation for neutrino kinetic equations. Part I: analysis of an idealized relaxation model. *Math. Comput.* **94**(353), 1199–1233 (2025)
56. Zhang, H., Yan, J., Qian, X., Song, S.: Numerical analysis and applications of explicit high order maximum principle preserving integrating factor Runge-Kutta schemes for allen-cahn equation. *Appl. Numer. Math.* **161**, 372–390 (2021)
57. Zhang, H., Yan, J., Qian, X., Song, S.: Up to fourth-order unconditionally structure-preserving parametric single-step methods for semilinear parabolic equations. *Comput. Methods Appl. Mech. Eng.* **393**, 114817 (2022)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.