MIT Libraries | DSpace@MIT

MIT Open Access Articles

A space-time adaptive method for reservoir flows: formulation and one-dimensional application

The MIT Faculty has made this article openly available. ***Please share*** how this access benefits you. Your story matters.

Citation: Jayasinghe, Savithru, et al. "A Space-Time Adaptive Method for Reservoir Flows: Formulation and One-Dimensional Application." Computational Geosciences, vol. 22, no. 1, Feb. 2018, pp. 107–23.

As Published: http://dx.doi.org/10.1007/s10596-017-9673-9

Publisher: Springer International Publishing

Persistent URL: http://hdl.handle.net/1721.1/114425

Version: Author's final manuscript: final author's manuscript post peer review, without publisher's formatting or copy editing

Terms of use: Creative Commons Attribution-Noncommercial-Share Alike

Massachusetts Institute of Technology

Computational Geosciences manuscript No.
(will be inserted by the editor)

A space-time adaptive method for reservoir flows: formulation and one-dimensional application

Savithru Jayasinghe · David L. Darmofal · Nicholas K. Burgess · Marshall C. Galbraith · Steven R. Allmaras

Received: date / Accepted: date

Abstract This paper presents a space-time adaptive framework for solving porous media flow problems, with specific application to reservoir simulation. A fully unstructured mesh discretization of space and time is used instead of a conventional time-marching approach. A space-time discontinuous Galerkin finite element method is employed to achieve a high-order discretization on the anisotropic, unstructured meshes. Anisotropic mesh adaptation is performed to reduce the error of a specified output of interest, by using *a posteriori* error estimates from the dual weighted residual method to drive a metric-based mesh optimization algorithm. The space-time adaptive method is tested on a one-dimensional two-phase flow problem, and is found to be more efficient in terms of computational cost (degrees-of-freedom and total runtime) required to achieve a specified output error level, when compared to a conventional first-order time-marching finite volume method and the space-time discontinuous Galerkin method on structured meshes.

Keywords unstructured space-time methods · anisotropic mesh adaptation · discontinuous Galerkin · high-order · two-phase flow

This research was supported through a Research Agreement with Saudi Aramco, a Founding Member of the MIT Energy Initiative (http://mitei.mit.edu/), with technical monitors Dr. Ali Dogru and Dr. Nicholas Burgess.

S. Jayasinghe · D. Darmofal · M. Galbraith · S. Allmaras
Massachusetts Institute of Technology, 77 Massachusetts Ave., Cambridge, MA
E-mail: savithru@mit.edu, darmofal@mit.edu, galbramc@mit.edu, allmaras@mit.edu

N. Burgess
Aramco Services Company, 400 Technology Square, Cambridge, MA
E-mail: nburgess3@gmail.com

1 Introduction

Numerical simulation has become an essential tool for analyzing and predicting the performance of reservoirs. In the context of hydrocarbon reservoirs in particular, computational fluid dynamics (CFD) models are used to investigate flow processes, assess the viability of different oil recovery methods, and ultimately predict the overall performance of the reservoir under different operating conditions. Results of these numerical simulations greatly influence engineering and management decisions, hence their accuracy and reliability are of significant importance.

A CFD model typically utilizes a mesh to discretize the domain of interest and approximates the flow solution on this mesh. The resolution of the mesh controls both the accuracy and the cost (e.g. degrees of freedom) of the numerical solution. Increasing the mesh resolution by adding more elements is a common approach to improve the solution fidelity. However, this approach is often limited by available computing power. Even with recent advances in parallel computing, the most powerful reservoir simulators are only just entering the regime of billion-cell models [17,34]. Finer meshes allow the CFD solution to capture features such as saturation fronts, gas breakthroughs and regions of trapped oil, all of which contribute to the performance of the reservoir. These prominent solution features often arise due to the multi-scale nature of the problems, heterogeneity of the geology, and the nonlinearity of the governing equations; hence, determining the size, location and orientation of these features beforehand is a non-trivial task. Therefore, the objective of this work is to develop an adaptive method that can autonomously modify the discrete mesh according to the nature of the solution, to produce a more reliable and accurate output.

1.1 Space-time methods

Typically, an unsteady partial differential equation (PDE) is first discretized in space to produce a set of ordinary differential equations that are then discretized in time, following what is often referred to as a method of lines approach. Most reservoir simulations use first or second order accurate temporal discretizations, such as the Backward Euler method [5,37,39]. However, an alternative is to apply the finite element method along the temporal axis as well. The idea of using this "space-time finite element method" dates back to the 1960s, to the work of Oden [35], Argyris and Scharpf [3], and Fried [21].

In the conventional approach, the same temporal discretization is applied to all the ordinary differential equations resulting from the spatial discretization, producing a *structured* space-time discretization. From a space-time perspective, this is equivalent to a tensor-product space-time mesh, where each space-time element is a tensor-product of a spatial element and a time-interval. However, as discussed in [23], the potential of the space-time finite element method lies in the use of *unstructured* space-time meshes, where arbitrarily oriented, anisotropic space-time elements can capture solution features more efficiently compared to more constrained tensor-product elements.

Hughes and Hulbert solved the second-order hyperbolic elastodynamic PDE using a space-time method with a continuous Galerkin (CG) method in space and a discontinuous Galerkin (DG) method in time [23, 24]. Their method partitions the space-time domain into decoupled time-slabs, which are solved sequentially by employing the solution at the end of the current time-slab as the initial condition for the next. However, they allow the space-time mesh to be unstructured within each time-slab, making their method attractive for space-time adaptive schemes. More recently, Chen et al. [13] developed a DG method in both space and time to solve a single-phase porous media flow problem using a quadrilateral (though unstructured) mesh. In both Hughes and Hulbert and Chen et al, a specific space-time adaptive algorithm is not proposed. In [45], Yano and Darmofal proposed a space-time DG method with fully-unstructured anisotropic mesh adaptation, and demonstrated that it can significantly improve the error-to-degrees-of-freedom efficiency of solving wave-propagation problems for one and two-dimensional spatial domains, compared to tensor-product space-time mesh adaptation. In this paper, we extend the Yano and Darmofal approach to porous media flows problems, specifically in the context of reservoir simulations. The demonstrations in this paper are one-dimensional spatial problems and therefore are small enough to be solved in a single space-time domain (i.e. a single time-slab). However, for the larger problems that will arise for two and in particular three-dimensional spatial domains, our fully-unstructured adaptive method can be applied on time-slabs as suggested by Hughes and Hulbert.

1.2 Solution adaptive methods

In this work, the space-time adaptive DG method of Yano and Darmofal is applied to two-phase porous media flow problems. A key feature of this method is output-based error estimation and mesh adaptation. The general outline of the output-based solution adaptation framework is depicted in Figure 1. The process begins with a problem statement, which includes the initial mesh, the PDE to be solved, boundary conditions, initial conditions, output function, desired error tolerance and typically a parameter to denote the amount of computational resources available (i.e. maximum number of CPU hours). The PDE is then solved on this initial mesh and the output error estimates are computed. If the error estimate is larger than the specified tolerance, the adaptation algorithm will utilize localized error estimates to generate a new mesh. The process is then repeated on the new adapted mesh until the output error estimates meet the tolerance criterion or the solver runs out of the allocated time.

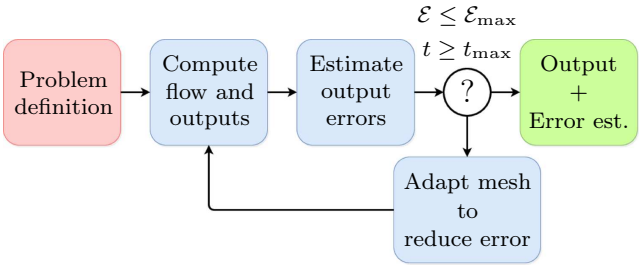

Fig. 1 General outline of adaptation framework

A variety of approaches exist for determining where adaptation should occur based upon the solution on the current mesh. For example, the magnitude of solution gradients can be used to identify important features [7, 12,14]. Another approach, based on the magnitude of the residual, has been demonstrated for porous media flows by Klieber using the DG method in [28] and by Amaziane et al. using the finite volume method in [1]. Our output-based adaptive method utilizes the dual-weighted residual (DWR) approach proposed by Becker and Rannacher [10,11] to obtain both global and local

error estimates, which are then used to drive the mesh adaptation.

This work focuses on h-adaptation, which involves changing the size and shape of elements in the mesh to control the total output error. A widely used strategy is to perform isotropic mesh refinement where selected elements are uniformly refined to decrease the error, as seen in [28,1,14] for flows through heterogeneous porous media. However, for problems involving highly anisotropic features, including the model reservoir flow applications in this work, anisotropic adaptation will be significantly more efficient. To combine output error estimates with anisotropic adaptation, we use the Mesh Optimization via Error Sampling and Synthesis (MOESS) algorithm proposed by Yano and Darmofal [45].

1.3 High-order methods

Reservoir simulations are often computed with *low-order* discretizations based on the finite volume method (FVM) [5,33] and finite difference methods (FDM) [37], where the term "low-order" typically refers to numerical methods that have at most second-order accuracy in space and time [43]. However, in recent years, high-order methods are being applied to porous media flow problems. Finite element methods, such as the DG method, offer a means to obtain high-order accurate solutions by increasing the order of the polynomial basis functions, and have been successfully applied to single phase, multiphase and linear transport flow problems [40,38,39]. Additionally, properties such as local mass conservation and ease of implementation on unstructured grids make the DG method a competitive alternative to the conventional low-order methods.

The use of a space-time DG discretization in this work also allows for high-order temporal discretizations, without being restricted to the first-order time-marching schemes that are largely used in practice. For smooth problems, the higher convergence rates allow high-order methods to achieve a given level of accuracy with fewer degrees of freedom compared to low-order methods [6]. However for problems with low regularity, the efficiency gains of high-order methods may not be realized without also utilizing mesh adaptation.

1.4 Applications to two-phase flow

The existing literature on the use of high-order DG finite element methods for solving two-phase flow problems is mostly focused on the decoupled pressure - saturation formulation. This technique, also known as the IMPES (implicit pressure, explicit saturation) method, is particularly attractive for incompressible flows because it allows the coupled system of equations to be separated into a purely elliptic "pressure" equation and an advection-dominated (nearly) hyperbolic "saturation" equation [5]. DG methods have been used to solve the pressure and saturation equations sequentially, as demonstrated by Rivière in [38] and Klieber in [28]. There also exist recent work where the pressure - saturation system is solved in a coupled, fully implicit manner, such as the work of Epshteyn in [20,19].

In this work, we abandon the sequential pressure-saturation approach and simultaneously solve the compressible mass conservation equations for each phase in a coupled and fully implicit manner. Solving the equations in mass conservation form avoids forming the global pressure equation, and is consistent with most industrial practices. The methods and results presented in this paper stem from the Master's thesis work of the first author [26].

1.5 Outline of paper

Section 2 introduces the compressible two-phase flow equations in mass conservation form, and gives a space-time formulation of the equations that is discretized using the space-time DG method described in Section 3. Section 4 briefly reviews the DWR method for output error estimation and the MOESS algorithm for mesh adaptation. Section 5 demonstrates the space-time adaptive framework on a 1D spatial test problem, and compares the adapted results with those from a conventional time-marching finite volume method, and the space-time DG method on structured meshes. Although the formulation introduced in Sections 2 - 4 are applicable to 3D spatial problems (and therefore 4D space-time problems), the application in Section 5 of this paper is to a 1D spatial problem (2D space-time problem). The conclusion section contains a brief discussion of the challenges involved in extending the proposed framework to multi-dimensional problems. Further, the computational cost of the entire mesh adaptation algorithm is compared with the cost of solving the primal PDE in Appendix B, showing that the proposed approach scales in a computationally feasible manner to multi-dimensional problems.

2 Governing equations for two-phase flow

The governing partial differential equations for two-phase flow are statements of mass conservation for each of the two phases in the porous medium, where the

non-wetting phase pressure (p_n) and the wetting phase saturation (S_w) are chosen to be the primal solution variables. The coupled, nonlinear system of equations under negligible gravitational effects is given by Eq. (1) below:

$$\frac{\partial}{\partial t}\left(\rho_w \phi S_w\right) - \nabla \cdot \left(\rho_w \frac{\mathbf{K} k_{rw}}{\mu_w}(\nabla p_n - \frac{\partial p_c}{\partial S_w}\nabla S_w)\right) = \rho_w q_w$$
$$\frac{\partial}{\partial t}\left(\rho_n \phi (1-S_w)\right) - \nabla \cdot \left(\rho_n \frac{\mathbf{K} k_{rn}}{\mu_n}\nabla p_n\right) = \rho_n q_n, \quad \forall \vec{x} \in \Omega_s,\ t \in I, \quad (1)$$

where ρ_w and ρ_n are the phase fluid densities (with subscripts w and n denoting the wetting and non-wetting phases, respectively), ϕ is the rock porosity, $\mathbf{K}$ is the rock permeability tensor, k_{rw} and k_{rn} are the relative permeability functions, μ_w and μ_n are the fluid viscosities, p_c is the capillary pressure, and q_w and q_n are source terms. The spatial domain is denoted by $\Omega_s \subset \mathbb{R}^d$, and $I = [0, T]$ is the time interval of interest. The underlying porous medium is assumed homogenous, i.e. the permeability field $\mathbf{K}$ is constant throughout the spatial domain. The following closure relations for the pressure and saturation variables are also required:

$$p_w + p_c = p_n \quad (2)$$
$$S_w + S_n = 1. \quad (3)$$

The constitutive relationships for the densities, porosity, relative permeabilities and capillary pressure are problem dependent, hence they are defined in Section 5 along with the description of the test problem.

2.1 Space-time formulation

The d-dimensional unsteady conservation laws in Eq. (1) are recast as $(d+1)$-dimensional conservation laws, yielding the following space-time formulation of the two-phase flow equations:

$$\sum_{j=1}^{d+1} \frac{\partial}{\partial \hat{x}_j}\hat{\mathbf{F}}_j^{\text{adv}}(\mathbf{u}) - \sum_{j=1}^{d+1} \frac{\partial}{\partial \hat{x}_j}\hat{\mathbf{F}}_j^{\text{diff}}(\mathbf{u}, \nabla \mathbf{u}) = \mathbf{S}(\mathbf{u}, \hat{\vec{x}}), \quad \forall \hat{\vec{x}} \in \Omega, \quad (4)$$

where $\mathbf{u} = [p_n, S_w]^T \in \mathbb{R}^{m=2}$ and $\hat{\vec{x}} = [\vec{x}, t] \in \mathbb{R}^{d+1}$ is the augmented space-time coordinate in the space-time domain $\Omega = \Omega_s \times I \subset \mathbb{R}^{d+1}$. The space-time advective flux $\hat{\vec{\mathbf{F}}}^{\text{adv}} \in \mathbb{R}^{m\times(d+1)}$, diffusive flux $\hat{\vec{\mathbf{F}}}^{\text{diff}} \in \mathbb{R}^{m\times(d+1)}$, and source term $\mathbf{S} \in \mathbb{R}^m$ are given by:

$$\hat{\mathbf{F}}_i^{\text{adv}}(\mathbf{u}) = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \quad \text{for } i = 1, .., d \quad (5)$$

$$\hat{\mathbf{F}}_{d+1}^{\text{adv}}(\mathbf{u}) = \begin{pmatrix} \rho_w \phi S_w \\ \rho_n \phi (1 - S_w) \end{pmatrix} \quad (6)$$

$$\hat{\mathbf{F}}_i^{\text{diff}}(\mathbf{u}, \hat{\nabla}\mathbf{u}) = \begin{pmatrix} \rho_w \frac{k_{rw}}{\mu_w}\mathbf{K}_{ij}(\frac{\partial p_n}{\partial \hat{x}_j} - \frac{\partial p_c}{\partial S_w}\frac{\partial S_w}{\partial \hat{x}_j}) \\ \rho_n \frac{k_{rn}}{\mu_n}\mathbf{K}_{ij}\frac{\partial p_n}{\partial \hat{x}_j} \end{pmatrix} \quad \text{for } i = 1, .., d \quad (7)$$

$$\hat{\mathbf{F}}_{d+1}^{\text{diff}}(\mathbf{u}, \hat{\nabla}\mathbf{u}) = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad (8)$$

$$\mathbf{S}(\mathbf{u}, \hat{\vec{x}}) = \begin{pmatrix} \rho_w q_w \\ \rho_n q_n \end{pmatrix}. \quad (9)$$

The diffusive flux is also assumed to be a linear function of $\hat{\nabla}\mathbf{u}$, and is decomposed as:

$$\hat{\vec{\mathbf{F}}}^{\text{diff}}(\mathbf{u}, \hat{\nabla}\mathbf{u}) = \hat{\vec{\mathbf{A}}}(\mathbf{u})\hat{\nabla}\mathbf{u}. \quad (10)$$

The boundary conditions are imposed using an operator $\mathcal{B}$ defined as:

$$\mathcal{B}(\mathbf{u}, \hat{\vec{\mathbf{F}}}^{\text{adv}}(\mathbf{u}) \cdot \hat{\vec{n}}, \hat{\vec{\mathbf{F}}}^{\text{diff}}(\mathbf{u}, \hat{\nabla}\mathbf{u}) \cdot \hat{\vec{n}}, \hat{\vec{x}}; BC) = 0, \quad \forall \hat{\vec{x}} \in \partial\Omega, \quad (11)$$

where $\hat{\vec{n}}$ is the space-time unit normal vector and BC represents the boundary condition data. The initial condition of the original unsteady conservation law is transformed by the above formulation into a Dirichlet boundary condition at the $t = 0$ boundary of the space-time domain Ω. This "temporal" boundary condition is implemented like any other spatial boundary condition using $\mathcal{B}$.

Note that in Eq. (4)-(11), *hat* accents have been used (i.e. $\hat{\nabla}(\cdot)$) to distinguish $(d+1)$-dimensional *space-time* vectors, fluxes and operators from their d - dimensional spatial counterparts. The rest of this paper assumes a space-time formulation, hence, the hat accents will be omitted for clarity.

3 Space-time DG discretization

The discontinuous Galerkin discretization seeks a solution in a finite dimensional function space $V_{h,p}$, which is defined as:

$$V_{h,p} \equiv \left\{\mathbf{v} \in [L^2(\Omega)]^m : \mathbf{v}|_\kappa \in [\mathcal{P}^p(\kappa)]^m, \forall \kappa \in \mathcal{T}_h\right\}. \quad (12)$$

$V_{h,p}$ represents the piecewise discontinuous solution space of p^{th}-order polynomials on each element of $\mathcal{T}_h$, where $\mathcal{T}_h$ is a triangulation of the space-time domain Ω into non-overlapping elements κ of characteristic size h.

Multiplying Eq. (4) by a test function $\mathbf{v}_{h,p} \in V_{h,p}$ and integrating by parts yields the weak formulation of the governing equation. Solving this weak formulation involves finding a solution $\mathbf{u}_{h,p} \in V_{h,p}$ that satisfies:

$$\mathcal{R}_{h,p}(\mathbf{u}_{h,p}, \mathbf{v}_{h,p}) = 0, \qquad \forall \mathbf{v}_{h,p} \in V_{h,p}, \tag{13}$$

where the semi-linear weighted residual $\mathcal{R}_{h,p} : V_{h,p} \times V_{h,p} \to \mathbb{R}$ is composed of three terms:

$$\begin{aligned}\mathcal{R}_{h,p}(\mathbf{u}_{h,p}, \mathbf{v}_{h,p}) = \mathcal{R}^{\text{adv}}_{h,p}(\mathbf{u}_{h,p}, \mathbf{v}_{h,p}) \\ + \mathcal{R}^{\text{diff}}_{h,p}(\mathbf{u}_{h,p}, \mathbf{v}_{h,p}) \\ + \mathcal{R}^{\text{source}}_{h,p}(\mathbf{u}_{h,p}, \mathbf{v}_{h,p}).\end{aligned} \tag{14}$$

$\mathcal{R}^{\text{adv}}_{h,p}(\mathbf{u}_{h,p}, \mathbf{v}_{h,p})$, $\mathcal{R}^{\text{diff}}_{h,p}(\mathbf{u}_{h,p}, \mathbf{v}_{h,p})$ and $\mathcal{R}^{\text{source}}_{h,p}(\mathbf{u}_{h,p}, \mathbf{v}_{h,p})$ represent the contributions of the advective, diffusive and source terms to the weighted residual, respectively.

3.1 Advective flux discretization

The DG discretization of the advective flux term is given by:

$$\begin{aligned}\mathcal{R}^{\text{adv}}_{h,p}(\mathbf{u}, \mathbf{v}) = &- \sum_{\kappa \in \mathcal{T}_h} \int_\kappa \nabla \mathbf{v}^T \cdot \vec{\mathbf{F}}^{\text{adv}}(\mathbf{u}) \, d\Omega \\ &+ \sum_{f \in \Gamma_I} \int_f (\mathbf{v}^+ - \mathbf{v}^-)^T \mathcal{H}(\mathbf{u}^+, \mathbf{u}^-; \vec{n}^+) \, d\Gamma \\ &+ \sum_{f \in \Gamma_B} \int_f {\mathbf{v}^+}^T \mathcal{H}^B(\mathbf{u}^+, \mathbf{u}^B(\mathbf{u}^+; BC); \vec{n}^+) \, d\Gamma,\end{aligned} \tag{15}$$

where $(\cdot)^+$ and $(\cdot)^-$ denote the trace values evaluated from opposite sides of a face f and $\vec{n}^+$ is the unit normal vector pointing from the $(+)$ side to the $(-)$ side of a face. Γ_I and Γ_B represent the set of interior and boundary faces, respectively. $\mathcal{H}$ and $\mathcal{H}^B$ are the numerical flux functions on the interior and boundary faces, respectively. In this work, $\mathcal{H}$ takes the form:

$$\mathcal{H}(\mathbf{u}^+, \mathbf{u}^-, \vec{n}^+) = \begin{cases} \mathbf{F}^{\text{adv}}_{d+1}(\mathbf{u}^+) \cdot n_t^+, & \text{if } n_t^+ \geq 0, \\ \mathbf{F}^{\text{adv}}_{d+1}(\mathbf{u}^-) \cdot n_t^+, & \text{otherwise,} \end{cases} \tag{16}$$

where n_t^+ denotes the temporal component of the space-time normal vector $\vec{n}^+$. The advective flux in the temporal direction is evaluated using the solution in the direction of decreasing time (i.e. in the past), in accordance with the laws of causality. At the domain boundaries, the numerical flux $\mathcal{H}^B$ is evaluated using a boundary state $\mathbf{u}^B$, which itself is a function of both the interior state $\mathbf{u}^+$ and the user-specified boundary condition data BC. For problems containing spatial advective fluxes, $\mathcal{H}$ and $\mathcal{H}^B$ need to be modified to perform an upwinding of the spatial fluxes as well [45,26].

3.2 Diffusive flux discretization

In most of the previous work where a pressure-saturation formulation of the two-phase flow equations is considered [38,28,20,19], the diffusive fluxes in the pressure equation are discretized using either the Oden-Baumann-Babuska (OBB) method [36], or a generalized form of the non-symmetric interior penalty Galerkin method (NIPG) [41], the symmetric interior penalty Galerkin method (SIPG) [4,44] and the incomplete interior penalty Galerkin method (IIPG) [16]. In this work, the diffusive flux terms are discretized using the second method proposed by Bassi and Rebay (BR2) [8,9]. For simplicity of notation, the jump $[\![\cdot]\!]$ and average $\{\cdot\}$ operators are defined for a scalar s and a vector $\vec{v}$ on an interior face as:

$$\begin{aligned}\{s\} &= \frac{1}{2}(s^+ + s^-), & \{\vec{v}\} &= \frac{1}{2}(\vec{v}^+ + \vec{v}^-) \\ [\![s]\!] &= s^+\vec{n}^+ + s^-\vec{n}^-, & [\![\vec{v}]\!] &= \vec{v}^+ \cdot \vec{n}^+ + \vec{v}^- \cdot \vec{n}^-\end{aligned} \tag{17}$$

The diffusive flux discretization can then be written as follows:

$$\begin{aligned}\mathcal{R}^{\text{diff}}_{h,p}(\mathbf{u}, \mathbf{v}) = & \\ &\sum_{\kappa \in \mathcal{T}_h} \int_\kappa \nabla \mathbf{v}^T \cdot \left(\vec{\mathbf{A}}(\mathbf{u}) \nabla \mathbf{u} \right) \, d\Omega \\ &- \sum_{f \in \Gamma_I} \int_f \left\{ \vec{\mathbf{A}}^T(\mathbf{u}) \nabla \mathbf{v} \right\}^T \cdot [\![\mathbf{u}]\!] \, d\Gamma \\ &- \sum_{f \in \Gamma_I} \int_f [\![\mathbf{v}]\!]^T \cdot \left\{ \vec{\mathbf{A}}(\mathbf{u}) \nabla \mathbf{u} \right\} \, d\Gamma \\ &- \sum_{f \in \Gamma_I} \int_f [\![\mathbf{v}]\!]^T \cdot \left\{ \vec{\mathbf{A}}(\mathbf{u}) \eta_f \vec{\mathbf{r}}_f([\![\mathbf{u}]\!]) \right\} \, d\Gamma \\ &- \sum_{f \in \Gamma_B} \int_f \left(\vec{\mathbf{A}}_B^T \nabla \mathbf{v}^+ \right)^T \cdot \left(\mathbf{u}^+ - \mathbf{u}^B \right) \cdot \vec{n}^+ \, d\Gamma \\ &- \sum_{f \in \Gamma_B} \int_f \left(\mathbf{v}^+ \vec{n}^+ \right)^T \cdot \vec{\mathbf{A}}_B \nabla \mathbf{u}^B \, d\Gamma \\ &- \sum_{f \in \Gamma_B} \int_f \left(\mathbf{v}^+ \vec{n}^+ \right)^T \cdot \vec{\mathbf{A}}_B \eta_f \vec{\mathbf{r}}_f((\mathbf{u}^+ - \mathbf{u}^B)\vec{n}^+) \, d\Gamma,\end{aligned} \tag{18}$$

where the boundary fluxes are set using $\mathbf{u}^B(\mathbf{u}^+; BC)$, $\vec{\mathbf{A}}_B(\mathbf{u}^B; BC)$, and $\nabla \mathbf{u}^B(\nabla \mathbf{u}^+; BC)$. The lifting operator $\vec{\mathbf{r}}_f : [V_{h,p}(f)]^{d+1} \to [V_{h,p}]^{d+1}$, essentially penalizes jumps in the solution across a face, and is defined as follows for an interior face f:

$$\sum_{\kappa \in \kappa_f} \int_\kappa \vec{\boldsymbol{\tau}}^T \cdot \vec{\mathbf{r}}_f(\vec{\mathbf{q}}) \, d\Omega = - \int_f \{\vec{\boldsymbol{\tau}}\}^T \cdot \vec{\mathbf{q}} \, d\Gamma, \tag{19}$$

$$\forall \vec{\boldsymbol{\tau}}, \vec{\mathbf{q}} \in [V_{h,p}]^{d+1}$$

where κ_f is the set of elements sharing the face f. For boundary faces, the lifting operator is defined as:

$$\int_{\kappa_B} \vec{\boldsymbol{\tau}}^T \cdot \vec{\mathbf{r}}_f(\vec{\mathbf{q}})\ d\Omega = -\int_f \vec{\boldsymbol{\tau}}^{+T} \cdot \vec{\mathbf{q}}\ d\Gamma, \qquad \forall \vec{\boldsymbol{\tau}}, \vec{\mathbf{q}} \in [V_{h,p}]^{d+1} \tag{20}$$

where κ_B is the element containing the boundary face. The stability of the DG discretization requires that the BR2 stabilization parameter, η_f, is greater than or equal to the number of faces in an element [22]. For the triangular and quadrilateral meshes used this work, η_f is set to values of 3 and 4 respectively.

3.3 Source discretization

The discretization of the source terms is as follows:

$$\mathcal{R}_{h,p}^{\text{source}}(\mathbf{u}, \mathbf{v}) = \sum_{\kappa \in \mathcal{T}_h} \int_\kappa \mathbf{v}^T \mathbf{S}(\mathbf{u}, \vec{x})\ d\Omega. \tag{21}$$

3.4 Solution method

Expressing the solution $\mathbf{u}_h$ and the test function $\mathbf{v}_h$ in terms of an element-wise discontinuous polynomial basis yields a discrete nonlinear system of equations, which is then solved using Newton's method with a line search algorithm, employing the UMFPACK [15] sparse direct solver. A more detailed discussion of the solution method is given in [26].

4 Output error estimation and mesh adaptation

4.1 Output error estimation

Let the exact value of the output of interest be denoted by:

$$J = \mathcal{J}(\mathbf{u}), \tag{22}$$

where $\mathcal{J} : V \to \mathbb{R}$ is the output functional of interest and $\mathbf{u} \in V$ is the exact solution to the governing PDE. This is usually expressed as an integral quantity over a surface, such as the mass flow across a boundary, or over a volume, such as the average pressure in the domain. Since the exact solution is not available, an approximation to the exact output can be computed using the discrete DG solution $\mathbf{u}_{h,p} \in V_{h,p}$ as:

$$J_{h,p} = \mathcal{J}_{h,p}(\mathbf{u}_{h,p}), \tag{23}$$

where $\mathcal{J}_{h,p} : V_{h,p} \to \mathbb{R}$ is the discrete output functional. The true error between the exact output and its approximation is given by:

$$\mathcal{E}_{true} = J - J_{h,p} = \mathcal{J}(\mathbf{u}) - \mathcal{J}_{h,p}(\mathbf{u}_{h,p}). \tag{24}$$

Since $\mathcal{E}_{true}$ cannot be directly computed in general, the goal of output error estimation is to approximate this true error in the output functional. In this work, the dual-weighted residual (DWR) method proposed by Becker and Rannacher [10,11] is used.

The DWR method represents the true output error as:

$$\mathcal{E}_{true} = \mathcal{J}(\mathbf{u}) - \mathcal{J}_{h,p}(\mathbf{u}_{h,p}) = -\mathcal{R}_{h,p}(\mathbf{u}_{h,p}, \boldsymbol{\psi}), \tag{25}$$

where $\boldsymbol{\psi} \in W \equiv V + V_{h,p}$ is the true adjoint solution that satisfies:

$$\overline{\mathcal{R}}'_{h,p}[\mathbf{u}, \mathbf{u}_{h,p}](\mathbf{w}, \boldsymbol{\psi}) = \overline{\mathcal{J}}'_{h,p}[\mathbf{u}, \mathbf{u}_{h,p}](\mathbf{w}), \qquad \forall \mathbf{w} \in W, \tag{26}$$

where $\overline{\mathcal{R}}'_{h,p}[\mathbf{u}, \mathbf{u}_{h,p}] : W \times W \to \mathbb{R}$ and $\overline{\mathcal{J}}'_{h,p}[\mathbf{u}, \mathbf{u}_{h,p}] : W \to \mathbb{R}$ are the mean-value linearizations defined as:

$$\overline{\mathcal{R}}'_{h,p}[\mathbf{u}, \mathbf{u}_{h,p}](\mathbf{w},\mathbf{v}) \equiv \int_0^1 \mathcal{R}'_{h,p}\left[(1-\theta)\mathbf{u} + \theta \mathbf{u}_{h,p}\right](\mathbf{w}, \mathbf{v})\ d\theta, \tag{27}$$

$$\overline{\mathcal{J}}'_{h,p}[\mathbf{u}, \mathbf{u}_{h,p}](\mathbf{w}) \equiv \int_0^1 \mathcal{J}'_{h,p}\left[(1-\theta)\mathbf{u} + \theta \mathbf{u}_{h,p}\right](\mathbf{w}) d\theta. \tag{28}$$

$\mathcal{R}'_{h,p}[\mathbf{z}](\cdot,\cdot)$ and $\mathcal{J}'_{h,p}[\mathbf{z}](\cdot)$ denote the Fréchet derivatives of $\mathcal{R}_{h,p}(\cdot,\cdot)$ and $\mathcal{J}_{h,p}(\cdot)$ with respect to the first argument, evaluated about $\mathbf{z}$.

The true output error may also be expressed using the definition of the mean-value linearized residual as

$$\mathcal{E}_{true} = -\overline{\mathcal{R}}'_{h,p}[\mathbf{u}, \mathbf{u}_{h,p}](\mathbf{u} - \mathbf{u}_{h,p}, \boldsymbol{\psi} - \boldsymbol{\psi}_{h,p}), \tag{29}$$

which shows that the true output error is a function of the error in the primal solution, $\mathbf{u} - \mathbf{u}_{h,p}$, as well as the error in the adjoint solution, $\boldsymbol{\psi} - \boldsymbol{\psi}_{h,p}$.

The true adjoint $\boldsymbol{\psi}$ is not computable in general since it lives in an infinite dimensional space W, and its computation requires the true primal solution. Hence, the true adjoint solution is approximated by a finite dimensional adjoint $\boldsymbol{\psi}_{h,\hat{p}} \in V_{h,\hat{p}}$ (for $\hat{p} > p$) which is obtained by solving a dual problem linearized about $\mathbf{u}_{h,p}$:

$$\mathcal{R}'_{h,\hat{p}}[\mathbf{u}_{h,p}](\mathbf{v}_{h,\hat{p}}, \boldsymbol{\psi}_{h,\hat{p}}) = \mathcal{J}'_{h,\hat{p}}[\mathbf{u}_{h,p}](\mathbf{v}_{h,\hat{p}}), \quad \forall \mathbf{v}_{h,\hat{p}} \in V_{h,\hat{p}}. \tag{30}$$

The DWR error estimate of the output is obtained by substituting this approximate adjoint into Eq. (25):

$$\mathcal{E}_{true} \approx -\mathcal{R}_{h,p}(\mathbf{u}_{h,p}, \boldsymbol{\psi}_{h,\hat{p}}). \tag{31}$$

The approximate adjoint $\boldsymbol{\psi}_{h,\hat{p}}$ needs to exist in a space that is richer than that of the approximate primal solution $\mathbf{u}_{h,p}$ (i.e. $V_{h,\hat{p}} \supset V_{h,p}$), else the DWR estimate yields zero due to Galerkin orthogonality. In this work, the polynomial order of the adjoint approximation is chosen to be one order higher than that of the primal solution, i.e. $\hat{p} = p + 1$.

A global estimate of the output error is not sufficient for mesh adaptation since it needs to identify regions in the domain with large and small contributions to the error. Therefore, a localized error estimate η_κ, associated with element κ, is obtained by an element-wise restriction of the adjoint weight as follows:

$$\eta_\kappa \equiv \left|\mathcal{R}_{h,p}(\mathbf{u}_{h,p}, \boldsymbol{\psi}_{h,\hat{p}}|_\kappa)\right|. \tag{32}$$

A bound of the error estimate can be obtained by summing the local error estimates over all elements:

$$\mathcal{E} \equiv \sum_{\kappa \in \mathcal{T}_h} \eta_\kappa. \tag{33}$$

4.2 Mesh adaptation

The MOESS algorithm used in this work relies on the duality between a continuous Riemannian metric field $\boldsymbol{\mathcal{M}} = \{\mathcal{M}(\vec{x})\}_{\vec{x}\in\Omega}$, consisting of $(d+1)\times(d+1)$ symmetric positive definite tensors $\mathcal{M}(\vec{x})$, and a discrete mesh $\mathcal{T}_h$. A mesh is said to be metric-conforming if all its edges are close to unit length as measured under the Riemannian metric field $\boldsymbol{\mathcal{M}}$. Following the work of Loseille and Alauzet [30], the mesh adaptation problem can be posed as a continuous optimization problem that seeks an optimal metric field $\boldsymbol{\mathcal{M}}^*$:

$$\boldsymbol{\mathcal{M}}^* = \arg\inf_{\boldsymbol{\mathcal{M}}} \mathcal{E}(\boldsymbol{\mathcal{M}}) \qquad \text{s.t.} \qquad \mathcal{C}(\boldsymbol{\mathcal{M}}) \le C_{\max}, \tag{34}$$

where $\mathcal{E}$ and $\mathcal{C}$ are the error and cost functionals, respectively. This work considers the cost to be the number of degrees of freedom in the solution, hence, the cost functional is defined as:

$$\mathcal{C}(\boldsymbol{\mathcal{M}}) = \int_\Omega c_p \sqrt{\det(\boldsymbol{\mathcal{M}}(\vec{x}))}\, d\vec{x}, \tag{35}$$

where c_p is the number of degrees of freedom (DOF) in the reference element, normalized by its size. $C_{\max}$ is the maximum DOF count, often set by the amount of available computational resources. Furthermore, the MOESS algorithm assumes that the total output error is the sum of elementwise local error contributions η_κ, and that each local contribution η_κ is also a function of the elemental metric tensor $\mathcal{M}_\kappa$. Hence, the error function $\mathcal{E}$ is approximated as:

$$\mathcal{E}(\boldsymbol{\mathcal{M}}) \approx \sum_{\kappa \in \mathcal{T}_h} \eta_\kappa(\mathcal{M}_\kappa). \tag{36}$$

Since the form of the local error function $\eta_\kappa(\mathcal{M}_\kappa)$ is not known *a priori*, surrogate models of these functions are constructed using the following sampling procedure: for all $\kappa_0 \in \mathcal{T}_h$,

1. Compute the elemental metric tensor $\mathcal{M}_{\kappa_0}$ and the local error estimate $\eta_{\kappa_0} \equiv |\mathcal{R}_{h,p}(\mathbf{u}_{h,p}, \boldsymbol{\psi}_{h,\hat{p}}|_{\kappa_0})|$.
2. Consider a set of local configurations, $\{\kappa_i\}_{i=1}^{n_{\text{config}}}$, each of which is obtained by splitting one or multiple edges of element κ_0. The solution on each new local configuration is computed via local solves, where the solution outside of the refined element is fixed. The new elemental metrics and local error estimates are computed for each local configuration and stored as pairs $\{\mathcal{M}_{\kappa_i}, \eta_{\kappa_i}\}_{i=1}^{n_{\text{config}}}$.
3. Fit the data in the metric-error pairs to a local error model of the form:

$$\eta_\kappa(\mathcal{M}_\kappa(S_\kappa)) = \eta_{\kappa_0} \exp(\text{tr}(R_\kappa S_\kappa)), \tag{37}$$

where S_κ is a $(d+1)\times(d+1)$ symmetric "step" tensor that is defined as:

$$S_\kappa(\mathcal{M}_\kappa) \equiv \log(\mathcal{M}_{\kappa_0}^{-\frac{1}{2}} \mathcal{M}_\kappa \mathcal{M}_{\kappa_0}^{-\frac{1}{2}}). \tag{38}$$

R_κ is a $(d+1)\times(d+1)$ symmetric "rate" tensor that is computed by performing the following least-squares regression on the data from the metric-error pairs:

$$R_\kappa = \arg\min_{Q \in Sym_{(d+1)}} \sum_{i=1}^{n_{\text{config}}} \left(f_{\kappa_i} - \text{tr}(Q S_{\kappa_i}(\mathcal{M}_{\kappa_i}))\right)^2, \tag{39}$$

where $f_{\kappa_i} = \log\left(\frac{\eta_{\kappa_i}}{\eta_{\kappa_0}}\right)$.

The step tensor S_κ characterizes the change in the metric from $\mathcal{M}_{\kappa_0}$ to $\mathcal{M}_\kappa$. Therefore, the rate tensor R_κ can be thought of as a generalization of the convergence rate for isotropic scaling to anisotropic changes.

A single iteration of the mesh adaptation algorithm detailed in [46] can be summarised as follows:

1. Given a mesh, determine the primal solution $\mathbf{u}_{h,p}$ and adjoint solution $\boldsymbol{\psi}_{h,\hat{p}}$.
2. Construct the local surrogate error models via local sampling.

3. Use a nonlinear optimization method to obtain the metric field that minimizes the error for a given maximum cost.
4. Generate a new mesh that conforms to the obtained metric field.

This work utilizes the globally-convergent method-of-moving-asymptotes (MMA) algorithm [42] implemented in NLopt [27] to solve the optimization problem. The adapted meshes are generated using a metric-conforming mesher developed by Loseille and Löhner [31,32].

5 Application

5.1 Test problem

The test problem considered in this work involves a 1D reservoir of length $L = 2000$ ft, containing a 1000-ft trapped oil region in the center, and aquifers on either side. A production well of length $L_s = 10$ ft is located at the center of the domain, extracting fluid according to a specified well model. The objective of the problem is to accurately compute the recovery factor of the reservoir over a production duration of $T = 1000$ days.

The governing equations are given by Eq. (1), with the initial condition set up to reflect the presence of the trapped oil and aquifer regions of the reservoir:

$$p_n(x,0) = 2500 \text{ psi}, \qquad \forall x \in [0, L] \tag{40}$$

$$S_w(x,0) = \begin{cases} 0.1, & 0.25L \le x \le 0.75L \\ 1.0, & x < 0.25L \text{ or } x > 0.75L. \end{cases} \tag{41}$$

The boundary conditions at the left and right ends of the reservoir are Dirichlet pressure and saturation conditions, which model an influx of water, replenishing the aquifers as the production well draws out fluid:

$$p_n(0,t) = p_n(L,t) = 2500 \text{ psi}, \qquad \forall t \in [0, T] \tag{42}$$

$$S_w(0,t) = S_w(L,t) = 1.0, \qquad \forall t \in [0, T]. \tag{43}$$

In particular, the boundary state $\mathbf{u}^B$ used in Eq. (15) and (18) is set from the BC data above, and the boundary state gradient $\nabla \mathbf{u}^B$ is set from the interior state gradient $\nabla \mathbf{u}^+$.

The production well is modeled by the source terms $q_w(x,t)$ and $q_n(x,t)$ which are defined for each phase as follows:

$$q_\alpha = -K \frac{k_{r\alpha}}{\mu_\alpha} \frac{p_n - p_b}{0.25 L_s^2} z(x), \qquad \text{for } \alpha = \{w, n\}, \tag{44}$$

where $z(x)$ is a smooth weighting function defined as:

$$z(x) = \begin{cases} 0, & 0 \le x \le 992.5 \\ 3\xi_1^2 - 2\xi_1^3, & 992.5 < x < 997.5 \\ 1, & 997.5 \le x \le 1002.5 \\ 1 - (3\xi_2^2 - 2\xi_2^3), & 1002.5 < x < 1007.5 \\ 0, & 1007.5 \le x \le 2000, \end{cases} \tag{45}$$

where,

$$\xi_1 = \frac{x - 992.5}{5}, \tag{46}$$

$$\xi_2 = \frac{x - 1002.5}{5}. \tag{47}$$

The volumetric flow rate of each phase at the extraction well depends on the specified bottomhole pressure p_b, and also on the solution variables p_n and S_w. Note that appropriate unit conversions are applied to the quantities given above to make the equations dimensionally consistent. The relevant constitutive relationships for densities, porosity, relative permeabilities and capillary pressure are given below:

$$\begin{aligned} \rho_\alpha &= \rho_{\alpha_{\text{ref}}}\, e^{c_\alpha (p_\alpha - p_{\text{ref}})} \qquad \text{for } \alpha \in \{w, n\}, \\ \phi &= \phi_{\text{ref}}\, e^{c_\phi (p_n - p_{\text{ref}})}, \\ k_{rw} &= S_w^2, \\ k_{rn} &= S_n^2 = (1 - S_w)^2, \\ p_c &= p_{c_{max}} (1 - S_w), \end{aligned}$$

where,

$$\begin{aligned} \phi_{\text{ref}} &= 0.3, & p_{\text{ref}} &= 14.7 \text{ psi}, \\ \rho_{w_{\text{ref}}} &= 62.4 \text{ lbf/ft}^3, & \rho_{n_{\text{ref}}} &= 52.1 \text{ lbf/ft}^3, \\ c_w &= 5 \times 10^{-6} \text{ psi}^{-1}, & c_n &= 1.5 \times 10^{-5} \text{ psi}^{-1}, \\ c_\phi &= 3 \times 10^{-6} \text{ psi}^{-1}, & K &= 200 \text{ md}, \\ \mu_w &= 1 \text{ cP}, & \mu_n &= 2 \text{ cP}, \\ p_{c_{max}} &= 5 \text{ psi}, & p_b &= 2350 \text{ psi}. \end{aligned}$$

The output functional of interest is the oil recovery factor:

$$J = \frac{Q_{n_{out}}}{V_{OIP}}, \tag{48}$$

where $Q_{n_{out}}$ is the total volume of oil extracted over T days, and V_{OIP} is the total volume of oil-in-place at $t = 0$, defined as follows:

$$Q_{n_{out}} = \int_0^T \int_0^L -q_n \, dx dt, \tag{49}$$

$$V_{OIP} = \int_0^L \phi(p_n(x,0))(1 - S_w(x,0)) dx = 272.0206 \text{ ft}. \tag{50}$$

5.2 Numerical results

This section presents the results of the two-phase flow problem defined above, solved using the space-time DG method on unstructured adapted meshes, as described in Section 3. The results are compared with those obtained using the space-time DG method on structured quadrilateral meshes, and also with a time-marching first-order finite volume method using a two-point flux approximation and a Backward-Euler temporal scheme. The specific nature of the structured meshes used for both space-time DG and time-marching FV is described in Appendix A.

Figures 2 and 3 are space-time plots of the non-wetting phase pressure and wetting phase saturation obtained from a piecewise quadratic (P2) adapted space-time DG solution, containing approximately 25,000 degrees of freedom (DOF) for each primary variable. The final adapted mesh after 25 adaptation iterations is overlaid over both plots. The pressure p_n starts from the constant initial condition of 2500 psi, but quickly transitions to a nearly piecewise linear profile due to the action of the production well at the center of the domain. The pressure profile is symmetric about the center of the domain and reaches a minimum pressure equal to the specified bottomhole pressure, $p_b = 2350$ psi, as expected. The pressure gradients set up by the production well cause two saturation fronts, originating from the initial S_w discontinuities at $x = 500$ ft and $x = 1500$ ft respectively, to propagate inwards at a constant speed towards the well. Figure 3 shows that the water breakthrough time (the time at which the two

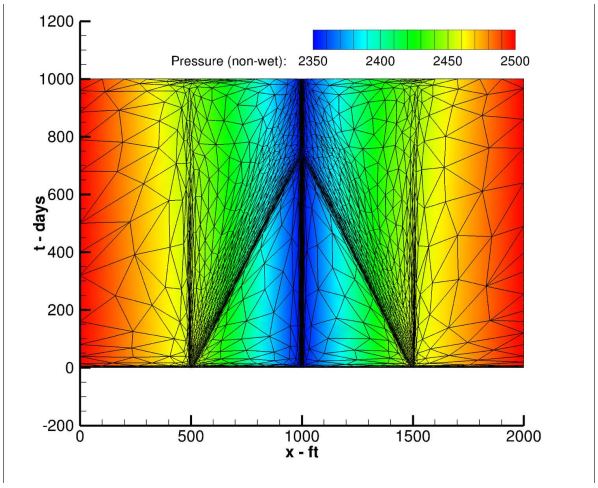

Fig. 2 Contour plot of non-wetting phase pressure (p_n) from a third-order (P2) adapted space-time DG solution with 25,000 DOF per primary variable (final mesh overlaid).

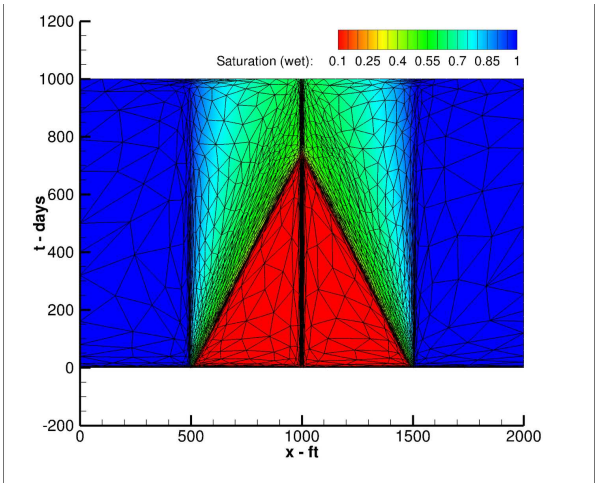

Fig. 3 Contour plot of wetting phase saturation (S_w) from a third-order (P2) adapted space-time DG solution with 25,000 DOF per primary variable (final mesh overlaid).

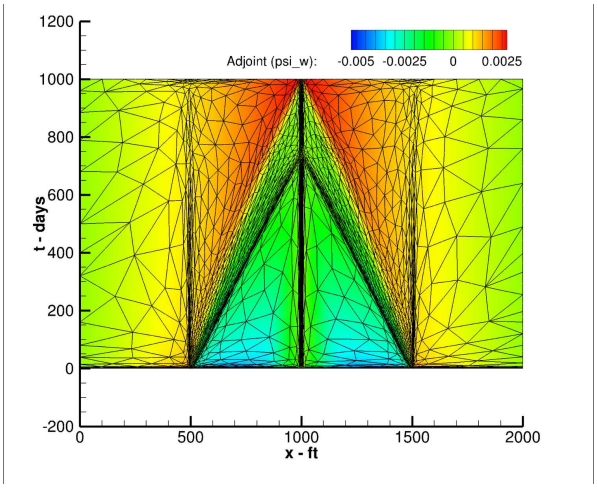

Fig. 4 Contour plot of a fourth-order (P3) adjoint ψ_w computed using a third-order (P2) adapted DG primal solution with 25,000 DOF per primary variable (final mesh overlaid).

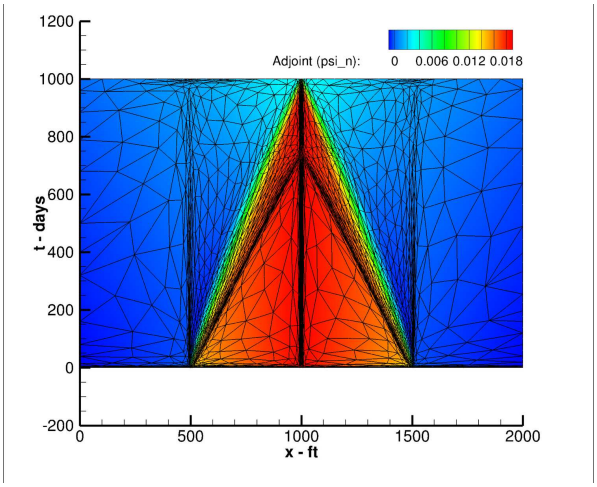

Fig. 5 Contour plot of a fourth-order (P3) adjoint ψ_n computed using a third-order (P2) adapted DG primal solution with 25,000 DOF per primary variable (final mesh overlaid).

combined rarefaction-shock waves intersect) is around $t = 750$ days.

Figures 4 and 5 show the behavior of the adjoint solutions, ψ_w and ψ_n, corresponding to each primal equation. The adjoint solutions are of order $\hat{p} = p + 1 = 3$, which are computed as a part of the error estimation process using the P2 primal solutions discussed above. Both adjoint plots contain a distinct triangular feature that is symmetric about the center of the domain. This triangular region can be identified to be the domain of influence for all adjoint characteristics that propagate backwards in time from the interior adjoint boundary condition along the $x = 1000$ ft line. Further insight into the nature of the adjoint solutions can be found in [25], which presents a theoretical analysis of the adjoint equations and boundary conditions for the compressible two-phase flow equations in mass conservation form.

Since the error estimates depend upon the approximation errors of both primal and adjoint solutions, the mesh adaptation algorithm focuses on regions where these errors are large. The dense anisotropic adaptation of the mesh along the production well, saturation fronts, bottom ($t = 0$) boundary, and also along the $x = 500$ ft and $x = 1500$ ft lines are consistent with this view, since the large third and higher order primal/adjoint derivatives present in those regions cannot be captured accurately by a piecewise quadratic solution without adaptation. The adaptation occuring along the bottom boundary is driven by a short initial pressure transient, arising from the parabolic nature of p_n, which is a result of the equations being slightly compressible. The zoomed-in view of p_n given in Figure 6 illustrates the initial pressure transient, where the elements in the final mesh are observed to be clustered in the temporal direction. Furthermore, Figures 7 - 9 contain simi-

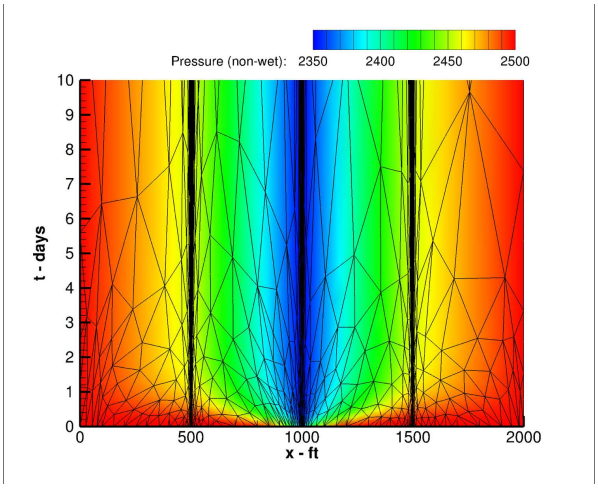

Fig. 6 Contour plot of p_n zoomed in on the first 10 days to show the initial transient behavior, with final mesh overlaid.

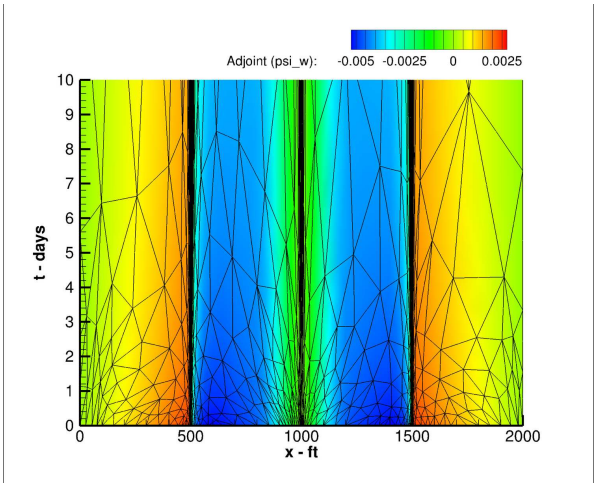

Fig. 8 Contour plot of ψ_w zoomed in on the first 10 days, with final mesh overlaid.

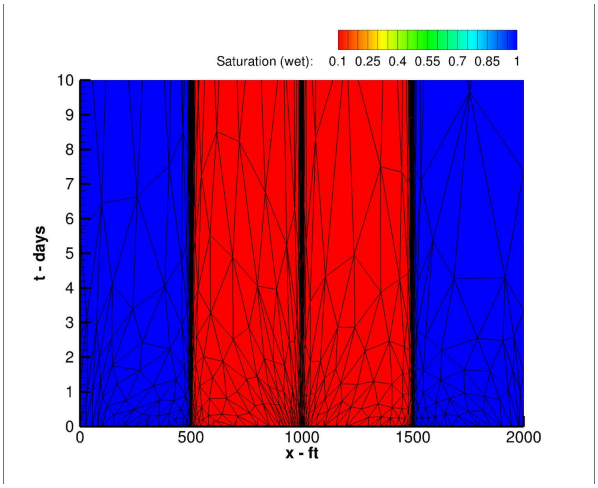

Fig. 7 Contour plot of S_w zoomed in on the first 10 days, with final mesh overlaid.

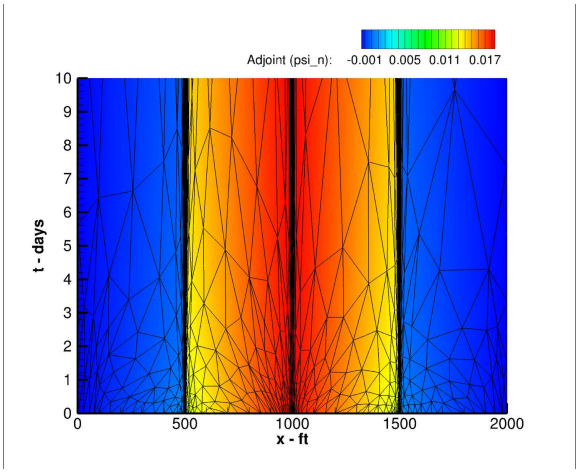

Fig. 9 Contour plot of ψ_n zoomed in on the first 10 days, with final mesh overlaid.

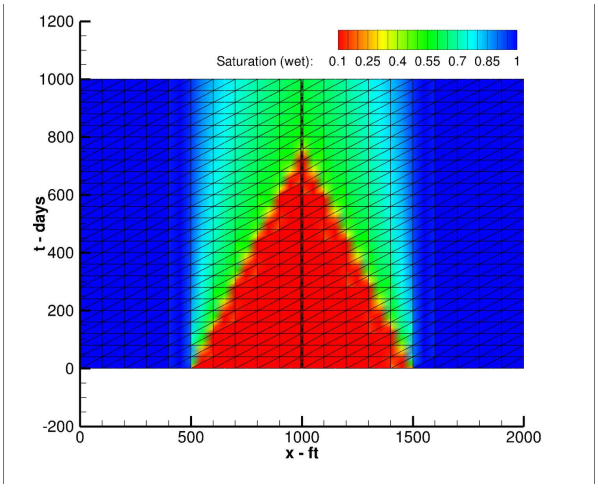

Fig. 10 Initial space-time mesh, with the P2 S_w solution.

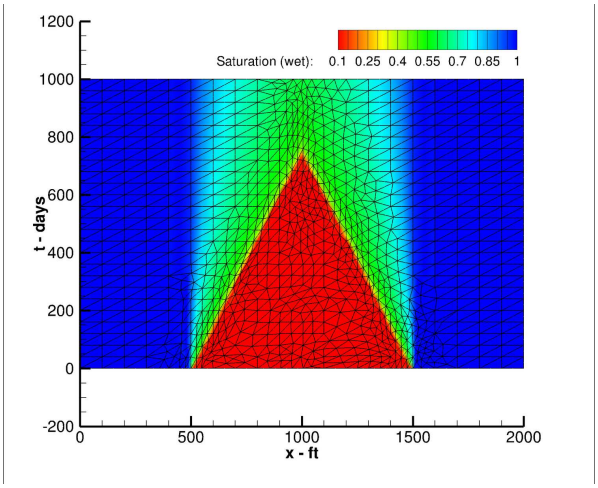

Fig. 11 Space-time mesh after 2 iterations of the adaptive algorithm, with the P2 S_w solution.

lar zoomed-in views of S_w, ψ_w and ψ_n respectively, but none of them exhibit a transient behavior.

Figure 10 shows the initial space-time mesh used for the adaptive process, with the corresponding P2 S_w solution. The initial mesh is structured, with a total of 22 spatial divisions (including 2 small divisions near the production well) and 25 temporal divisions. The intermediate space-time meshes obtained after 2, 8 and 15 iterations of the adaptive algorithm are shown in Figures 11, 12 and 13, respectively. These figures show that the adaptive algorithm immediately improves the mesh resolution in the vicinity of the production well (i.e. around the $x = 1000$ ft line) and the saturation fronts, since those regions are the most sensitive to the errors in the output. This is also reflected in Figure 14, which shows the history of the output error estimate, where a sharp decrease in output error is observed in the first few iterations. Changes to the mesh

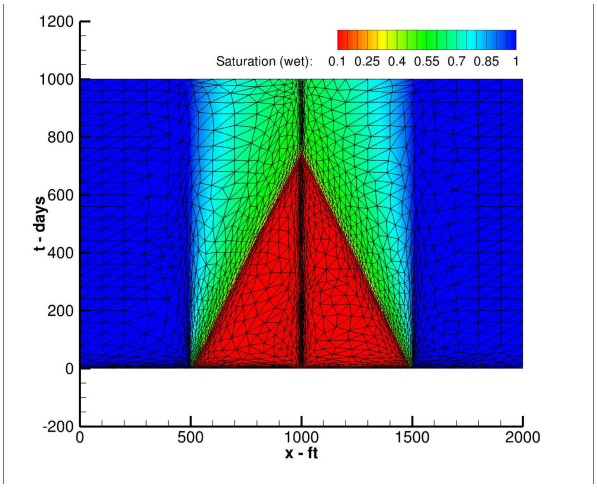

Fig. 12 Space-time mesh after 8 iterations of the adaptive algorithm, with the P2 S_w solution.

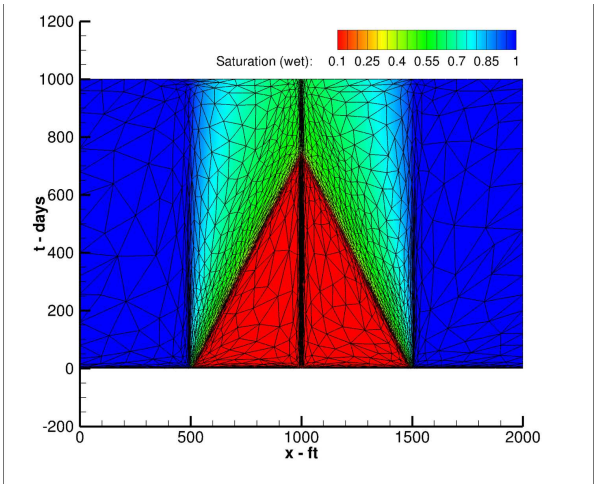

Fig. 13 Space-time mesh after 15 iterations of the adaptive algorithm, with the P2 S_w solution.

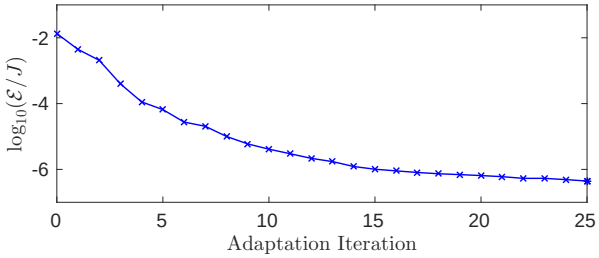

Fig. 14 Adaptation history of the output error estimate.

become smoother and subtler towards the latter part of the adaptive process, where most of the work involves coarsening the mesh in unimportant regions (e.g. aquifier regions at each end) so that more DOFs can be allocated to where needed. The error estimate curve in Figure 14 flattens out to a value that is about 4 orders of magnitude smaller than the initial error, as the mesh approaches its optimal configuration.

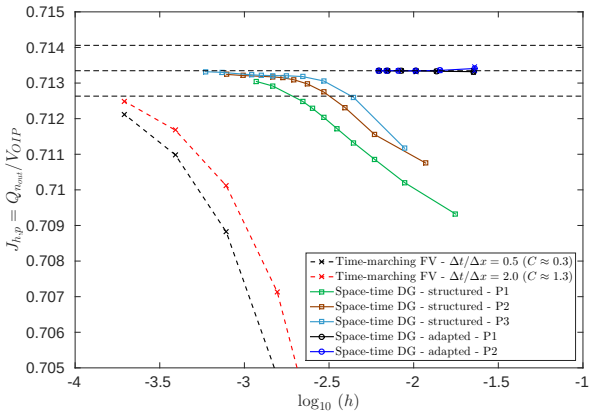

Fig. 15 Output J vs. h comparison between the different discretizations.

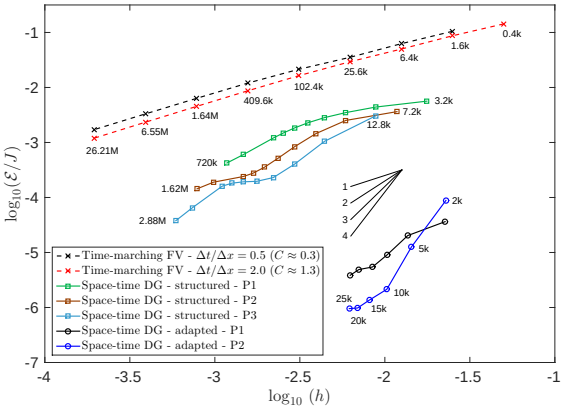

Fig. 16 Output error $\mathcal{E}$ vs. h comparison between the different discretizations.

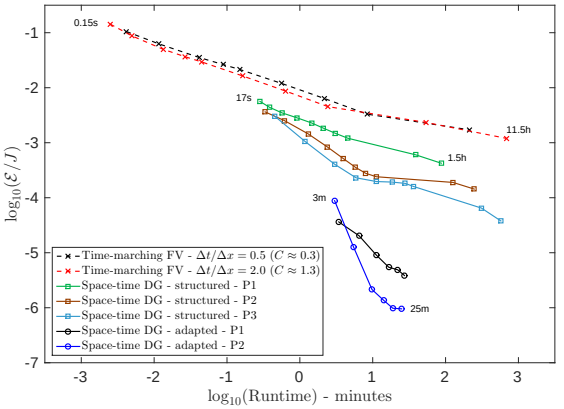

Fig. 17 Output error $\mathcal{E}$ vs. runtime comparison between the different discretizations.

Figure 15 shows a comparison of the oil recovery factors ($J_{h,p}$) predicted by the space-time DG method on adapted and structured meshes as well as the finite volume method on structured meshes. For the purpose of comparing the fidelity of the different solutions, a dimensionless average mesh size h is defined as follows:

$$h = \sqrt{\frac{1}{N}}, \tag{51}$$

where N is the total number of space-time degrees of freedom in the solution. The high-order space-time DG results (solid lines) converge to the exact functional value as h is decreased, while also incurring much smaller errors for a given h value, compared to the first-order finite volume results (dashed lines). The adapted DG results are well within the $\pm 0.1\%$ error interval marked by the dashed horizontal lines, even for the coarsest meshes. The coarsest space-time DG result on structured quadrilateral meshes has an output error of about 0.5%, but reaches $\pm 0.1\%$ after a few uniform mesh refinements. Although the FV meshes contain the same cell spacing distribution as the space-time structured meshes in the x-direction, the FV method requires many more refinements to achieve comparable levels of accuracy.

The ratio of spatial to temporal spacing has an impact on the efficiency of the structured mesh calculations. This spacing ratio is characterized using the CFL number C (see Appendix A for the specific definition used). The two FV curves in Figure 15 illustrate this, where the $C \approx 1.3$ refinement curve (red dashed line) represents a near-optimal behavior, as described in Appendix A.

Figure 16 compares how the output error, $\mathcal{E} \equiv J - J_{h,p}$, behaves with the mesh size h for the time-marching FV method, space-time DG method on uniformly refined structured meshes, and the space-time DG method with output-based mesh adaptation. Since the true output, J, is not known analytically, it is estimated using a space-time adapted DG solution containing approximately 50,000 degrees of freedom for each primary variable. The number next to each data point on the figure represents the total number of space-time DOF for that particular solution. The finite volume results, shown by the dashed lines, have first order convergence as expected. For a given value of h, the high-order space-time method on structured meshes achieves lower errors in comparison to FV, while also exhibiting higher convergence rates. The error convergence rate increases with the polynomial order of the solution as expected, but both P2 (brown) and P3 (light blue) curves contain plateau regions within which the error convergence rate is locally small. The space-time DG adaptive results (solid black and blue lines) are clearly the most efficient, since they consistently achieve errors that are at least two orders of magnitude smaller than the structured DG spacetime results, for the same h.

Figure 17 shows the total time (in minutes) taken by each method to achieve a given output error $\mathcal{E}$. The runtimes given for the space-time adaptive method include the time taken to generate multiple intermediate meshes, solve the primal and adjoint problems, and also perform error estimation. The FV and structured space-time results only represent the times taken to compute the primal solution and output. All simulations were run as single thread processes on a computer with an Intel i7-5930K (3.5 GHz) processor, 15MB cache and 32GB of RAM. The runtime results closely reflect the trends seen previously in Figure 16, with the space-time adaptive method being the most efficient by achieving orders of magnitude smaller errors in comparison to the FV and space-time structured results, for a given amount of computational time.

These results show that although high-order discretizations, particularly in the temporal direction, have an advantage over low-order methods such as the FV scheme used here, their benefits are fully realized only when coupled with mesh adaptation.

6 Conclusion

This paper presented an output-based mesh adaptation framework for solving a general unsteady conservation law, using a space-time discontinuous Galerkin formulation. This adaptive method was applied to the compressible two-phase flow equations in mass conservation form, with the oil recovery factor as the output of interest. Grid convergence studies were performed and the adaptive results were compared with results generated by a conventional 1D time-marching finite volume method with first order spatio-temporal accuracy, and a high-order space-time DG method on structured meshes.

The space-time adaptive method consistently achieved output errors that were orders of magnitude smaller compared with the other two methods, while using the least number of degrees of freedom and computational time of all three methods tested.

Although this paper only demonstrates a 1D application, the formulation is applicable to arbitrary spatial dimensions, and our current work is focused on 2D and then 3D applications. The simplified cost analysis presented in Appendix B shows that the overhead of performing h-adaptation relative to the primal solve is expected to decrease as the problem becomes more complex and multi-dimensional. Extending the framework to 2D spatial problems with simple domains would be relatively straightforward, since the generation of 3D space-time meshes at each adaptation iteration can be performed with existing 3D metric-based meshing software. However, the 4D meshing capabilities required to solve 3D spatial problems is largely an uncharted domain. Furthermore, the multi-dimensional extension of this work will require or greatly benefit from the development of more efficient solution methods (e.g. preconditioners) for solving the large nonlinear systems arising from the space-time DG discretizations.

A Optimal finite volume results

Figure 18 shows the space-time mesh corresponding to the coarsest FV case in Figure 16, where a symmetric, non-uniform cell spacing is used in the x-direction to allocate a larger fraction of the cells near the center of the domain. The mesh has 40 cells in the spatial direction, resulting in an average cell length of $\Delta x = 50$ ft. The lengths of the cells in the right half of the domain, between $x = 1000$ ft and $x = 2000$ ft, are given by:

$$\Delta x_i = \begin{cases} 7.5 \times 1.19073^i, & \text{for } 0 \leq i \leq 14 \\ 100, & \text{for } 15 \leq i \leq 19. \end{cases} \tag{52}$$

The spatial cell distribution specified above is also used to generate the coarsest structured space-time mesh for DG in Figure 16. In the temporal direction, the mesh uses a constant time-step Δt corresponding to a CFL number $C \approx 1.3$. All finer structured meshes are uniform refinements of the coarsest one.

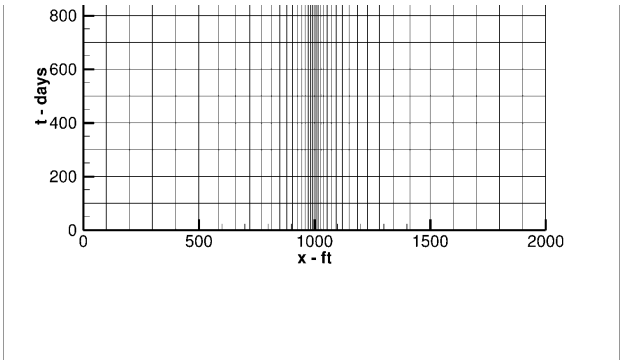

Fig. 18 A structured space-time mesh at a near-optimal CFL number, $C \approx 1.3$ ($\Delta x = 50$ ft, $\Delta t = 100$ days).

The first order spatial and temporal accuracy of the time-marching FV method used in this work implies that the FV errors can be modeled using the following relation:

$$\mathcal{E} = \alpha \frac{\Delta x}{L} + \beta \frac{\Delta t}{T}, \tag{53}$$

where Δx is the average length of FV cells, Δt is the size of the time-step, L and T are the domain sizes, and α and β are constants to be determined. Next, a CFL number C is defined as follows:

$$C = \frac{U \Delta t}{\Delta x}, \tag{54}$$

where U is a characteristic speed based on Darcy's equation for the wetting phase:

$$U = -\frac{1}{\phi}\frac{K}{\mu_w}\frac{\Delta p}{0.5L} = 0.6328 \text{ ft/day}, \tag{55}$$

and the pressure drop from the left boundary to the production well is $\Delta p = -150$ psi. Using the definition of h given by Eq. (51), and the definition of C from above, Eq. (53) is rewritten as:

$$\mathcal{E} = z(C) \cdot h, \tag{56}$$

where,

$$z(C) = \left(\frac{\alpha}{\sqrt{C}} + \beta\frac{L}{UT}\sqrt{C}\right)\sqrt{\frac{UT}{L}}. \tag{57}$$

The optimal error vs. h curve is one that minimizes $z(C)$, thus producing the minimum error for a fixed value of h. A series of FV simulations with different Δx and Δt combinations were computed, and linear regression was performed on the resulting errors to produce the following estimates for α and β:

$$\alpha = 2.3614 \tag{58}$$
$$\beta = 0.5492. \tag{59}$$

The optimal CFL number C^* is found by minimizing $z(C)$, yielding:

$$C^* = \frac{\alpha UT}{\beta L} \approx 1.36, \tag{60}$$

which corresponds to a $\frac{\Delta t}{\Delta x}$ ratio ≈ 2.1. However, for purposes of running FV simulations, a slightly sub-optimal $\frac{\Delta t}{\Delta x}$ ratio of 2 is chosen. Figure 19 shows the output error vs. h curves for a series of different FV simulations, where either Δx is fixed and Δt is refined (solid lines), or Δx and Δt are refined together at a fixed CFL number (dashed lines). The $C \approx 1.3$ curve is observed to lie along the optimal front of the meshes producing the lowest error for a given h.

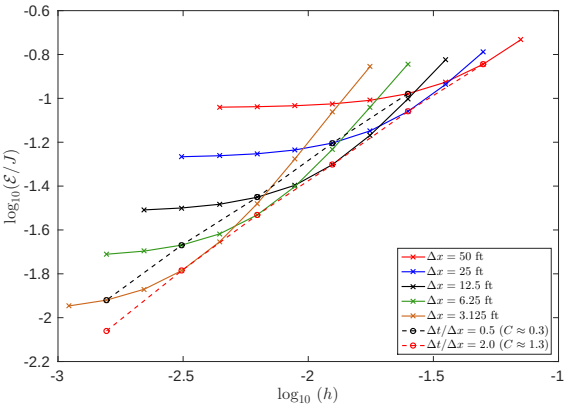

Fig. 19 Output error $\mathcal{E}$ vs. h for time-marching FV method

B Cost analysis

In the interest of fairly comparing computational costs between different spatial dimensions d, the mesh resolution is assumed to be fixed and characterized by an average mesh size $h < 1$ in each dimension, with the volume of the space-time domain being equal to 1. The number of $(d+1)$-simplex elements in the space-time mesh is given by:

$$N_{\text{elem}} \sim \frac{1}{V_{\text{simplex}}(d+1) \cdot h^{d+1}}, \tag{61}$$

where $V_{\text{simplex}}(d)$ represents the volume of a unit d-dimensional regular simplex:

$$V_{\text{simplex}}(d) = \frac{1}{d!}\sqrt{\frac{d+1}{2^d}}. \tag{62}$$

For a space-time DG discretization, the number of degrees of freedom per element, M, is given by:

$$M(d,p) = \frac{(p+d+1)!}{p!\ (d+1)!}\ N_{\text{state}}, \tag{63}$$

where p is the polynomial order of the DG solution and N_{state} is the number of state variables. Therefore, the total number of degrees of freedom on a space-time mesh with N_{elem} elements is given by:

$$N(d,p) = M(d,p) \cdot N_{\text{elem}}, \tag{64}$$

For simplicity, it is assumed that the complexity of the linear solver can be modeled as $\mathcal{O}(kM^3N_{\text{elem}}^r)$, where k is a constant that captures the nonlinearity and the conditioning of the physical problem. Highly nonlinear, poorly conditioned problems result in larger k values. The cubic power on M is a result of the dense coupling of DOFs within each element, which produces a dense matrix block of size $M \times M$ that needs to be solved for each element. However, since the DG discretization has sparse interactions between elements, the Jacobian matrix has a block sparse structure that can be exploited by sparse matrix solvers to produce a more efficient scaling on N_{elem}. Thus, the exponent r typically takes values in the range $1 \leq r \leq 2$ for well preconditioned iterative solvers of sparse systems. Therefore, the cost of solving the primal problem is given by:

$$C_{\text{primal}} = \mathcal{O}\left(k_{\text{primal}} \cdot (M(d,p))^3 \cdot (N_{\text{elem}})^r\right). \tag{65}$$

Similarly, the cost of solving the adjoint problem in the richer $(p+1)$ space is:

$$C_{\text{adjoint}} = \mathcal{O}\left(k_{\text{adjoint}} \cdot (M(d,p+1))^3 \cdot (N_{\text{elem}})^r\right) \tag{66}$$
$$= \mathcal{O}\left(k_{\text{adjoint}} \cdot \left(M(d,p) \cdot \frac{p+d+2}{p+1}\right)^3 \cdot (N_{\text{elem}})^r\right) \tag{67}$$

$$\frac{C_{\text{adjoint}}}{C_{\text{primal}}} \sim \frac{k_{\text{adjoint}}}{k_{\text{primal}}} \cdot \left(\frac{p+d+2}{p+1}\right)^3. \tag{68}$$

Although the cubic term in Eq. (68) suggests that the adjoint solve may be more expensive compared to the primal solve, the linearity of the adjoint problem compensates for this via the $k_{\text{adjoint}}/k_{\text{primal}}$ ratio, often making the adjoint solve cheaper than solving the nonlinear primal problem. The relative cost of the adjoint solve further diminishes with increasing p.

In the local sampling step of the MOESS algorithm, each local configuration is obtained by splitting an edge of an element to produce two sub-elements. The cost of performing a single local solve to compute the solution on these two sub-elements is given by:

$$C_{\text{config}} = \mathcal{O}\left(k_{\text{local}} \cdot (M(d,p))^3\right). \tag{69}$$

Therefore, the cost of all local solves is given by:

$$C_{\text{local}} = C_{\text{config}} \cdot N_{\text{config}} \cdot N_{\text{elem}}, \tag{70}$$

where N_{config} is the number of split configurations per element, which is equal to the number of edges in the $(d+1)$-simplex element:

$$N_{\text{config}} = \frac{1}{2}(d+1)(d+2). \tag{71}$$

Hence, the cost of local solves simplifies to:

$$C_{\text{local}} = \mathcal{O}\left(k_{\text{local}} \cdot (M(d,p))^3\right) \cdot \frac{1}{2}(d+1)(d+2) \cdot N_{\text{elem}} \tag{72}$$

$$= \mathcal{O}\left(k_{\text{local}} \cdot (M(d,p))^3 \cdot (N_{\text{elem}})^r\right) \cdot (d+1)(d+2) \cdot (N_{\text{elem}})^{1-r} \tag{73}$$

$$\frac{C_{\text{local}}}{C_{\text{primal}}} \sim \frac{k_{\text{local}}}{k_{\text{primal}}} \cdot (d+1)(d+2) \cdot \left(V_{\text{simplex}}(d+1) \cdot h^{d+1}\right)^{r-1} \tag{74}$$

If $r > 1$, the exponential decrease of $V_{\text{simplex}}(d+1) \cdot h^{d+1}$ with d ensures that the cost of all the local solves is cheaper compared to the primal solve at higher dimensions. Furthermore, each of the local problems are generally less nonlinear and better conditioned than the global problem. This is also observed in practice, where the primal solve typically takes $\mathcal{O}(10)$ nonlinear iterations whereas each local solve takes only $\mathcal{O}(1)$ nonlinear iterations. Therefore, even for the worst-case of $r = 1$ and $d = 3$, the $(d+1)(d+2)$ factor in Eq. (74) can be compensated by the $k_{\text{local}}/k_{\text{primal}}$ ratio, making the local sampling procedure cheaper than the primal solve.

The cost of generating a space-time mesh may be approximated by the complexity of computing a Delaunay triangulation of n points in d+1 dimensions, which is known to be $\mathcal{O}(n)$ in the expected case and $\mathcal{O}(n^{\lceil (d+1)/2 \rceil})$ in the worst-case [2, 18]. The relationship between n and N_{elem} for an isotropic space-time mesh can be approximated by the following relation:

$$n \approx \xi(d+1) \cdot N_{\text{elem}}, \tag{75}$$

where $\xi(d)$ is ratio between the number of vertices per d-simplex and the average number of regular d-simplices around a vertex, $S(d)$,

$$\xi(d) = \frac{d+1}{S(d)}. \tag{76}$$

Under the assumption of an isotropic mesh, a good approximation for $S(d)$ is the ratio:

$$S(d) \approx \frac{\Theta_{\text{sphere}}(d)}{\Theta_{\text{vertex}}(d)}, \tag{77}$$

where $\Theta_{\text{sphere}}(d)$ is the solid angle subtended by the surface of the d-dimensional unit ball at the origin, and $\Theta_{\text{vertex}}(d)$ is the solid angle subtended by a face of the regular d-simplex at its opposite vertex. Using the formula for the solid angular content at each vertex of a regular simplex given in [29], the above ratio can be written as:

$$\frac{\Theta_{\text{sphere}}(d)}{\Theta_{\text{vertex}}(d)} = \frac{2^d}{d!\; F_d\left(\frac{1}{2}\sec^{-1}(d)\right)}, \tag{78}$$

where $F_d(\alpha)$ is the recursive Schläfli function defined in Section 7.2 of [47]. Table 1 contains evaluations of the above ratios up to $d = 4$.

d	$S(d)$	$\xi(d)$
1	2	1
2	6	0.5
3	22.795	0.175
4	102.200	0.049

Table 1 Numerical values of $S(d)$ and $\xi(d)$

By assuming the expected linear complexity of the Delaunay triangulation, the cost of mesh generation relative to the primal solve is given by:

$$C_{\text{mesh}} = \mathcal{O}\left(\xi(d+1) \cdot N_{\text{elem}}\right) \tag{79}$$

$$\frac{C_{\text{mesh}}}{C_{\text{primal}}} \sim \frac{\xi(d+1)}{k_{\text{primal}} \cdot (M(d,p))^3 \cdot (N_{\text{elem}})^{r-1}} \tag{80}$$

$$\frac{C_{\text{mesh}}}{C_{\text{primal}}} \sim \frac{\xi(d+1) \cdot \left(V_{\text{simplex}}(d+1) \cdot h^{d+1}\right)^{r-1}}{k_{\text{primal}} \cdot (M(d,p))^3} \tag{81}$$

If $r > 1$, the decrease of the h^{d+1} term in Eq. (81) dominates (since $h < 1$), and causes C_{mesh} to be smaller relative to C_{primal} as d increases. The ratio $\xi(d+1)/M(d,p)^3$ also decreases with increasing d. Thus, even for an optimally scaling primal solver (i.e., $r = 1$), the mesh generation cost is a decreasing fraction of the primal solve cost as d increases.

Acknowledgements The authors wish to thank Dr. Eric Dow for reviewing this paper.

References

1. Amaziane, Brahim, Bourgeois, Marc, El Fatini, Mohamed: Adaptive Mesh Refinement for a Finite Volume Method for Flow and Transport of Radionuclides in Heterogeneous Porous Media. Oil Gas Sci. Technol. - Rev. IFP Energies nouvelles **69**(4), 687–699 (2014)
2. Amenta, N., Attali, D.: Abstract complexity of delaunay triangulation for points on lower-dimensional polyhedra (2007)
3. Argyris, J., Scharpf, D.: Finite elements in time and space. Nuclear Engineering and Design **10**(4), 456 – 464 (1969)
4. Arnold, D.N.: An interior penalty finite element method with discontinuous elements. SIAM Journal on Numerical Analysis **19**, 742–760 (1982)
5. Aziz, K., Settari, A.: Petroleum Reservoir Simulation. Elsevier (1979)
6. Babuska, I., Szabo, B.A., Katz, I.N.: The p-version of the finite element method. SIAM Journal on Numerical Analysis **18**(3), 515–545 (1981)

7. Baker, T.J.: Mesh adaptation strategies for problems in fluid dynamics. Finite Elements Anal. Design **25**, 243–273 (1997)
8. Bassi, F., Rebay, S.: GMRES discontinuous Galerkin solution of the compressible Navier-Stokes equations. In: K. Cockburn, Shu (eds.) Discontinuous Galerkin Methods: Theory, Computation and Applications, pp. 197–208. Springer, Berlin (2000)
9. Bassi, F., Rebay, S.: Numerical evaluation of two discontinuous Galerkin methods for the compressible Navier-Stokes equations. International Journal for Numerical Methods in Fluids **40**, 197–207 (2002)
10. Becker, R., Rannacher, R.: A feed-back approach to error control in finite element methods: Basic analysis and examples. East-West Journal of Numerical Mathematics **4**, 237–264 (1996)
11. Becker, R., Rannacher, R.: An optimal control approach to a posteriori error estimation in finite element methods. In: A. Iserles (ed.) Acta Numerica. Cambridge University Press (2001)
12. Cao, W., Huang, W., Russell, R.D.: Comparison of two-dimensional r-adaptive finite element methods using various error indicators. Mathematics and Computers in Simulation **56**(2), 127 – 143 (2001). Method of lines
13. Chen, Z., Steeb, H., Diebels, S.: A space-time discontinuous Galerkin method applied to single-phase flow in porous media. Computational Geosciences **12**(4), 525–539 (2008)
14. Chueh, C., Secanell, M., Bangerth, W., Djilali, N.: Multilevel adaptive simulation of transient two-phase flow in heterogeneous porous media. Computers & Fluids **39**(9), 1585 – 1596 (2010)
15. Davis, T.A.: Algorithm 832: UMFPACK V4.3 - An unsymmetric-pattern multifrontal method. ACM Transactions on mathematical software **30**(2), 196–199 (2004)
16. Dawson, C., Sun, S., Wheeler, M.F.: Compatible algorithms for coupled flow and transport. Computer Methods in Applied Mechanics and Engineering **193**(23-26), 2565 – 2580 (2004)
17. Dogru, A.H., Fung, L.S., Middya, U., Al-Shaalan, T.M., Pita, J.A.: A Next-Generation Parallel Reservoir Simulator for Giant Reservoirs. Society of Petroleum Engineers (2009)
18. Dwyer, R.A.: Higher-dimensional voronoi diagrams in linear expected time. Discrete & Computational Geometry **6**(3), 343–367 (1991)
19. Epshteyn, Y., Rivière, B.: On the solution of incompressible two-phase flow by a p-version discontinuous Galerkin method. Communications in Numerical Methods in Engineering **22**(7), 741–751 (2006)
20. Epshteyn, Y., Rivière, B.: Analysis of *hp* discontinuous Galerkin methods for incompressible two-phase flow. Journal of Computational and Applied Mathematics **225**(2), 487 – 509 (2009)
21. Fried, I.: Finite-element analysis of time-dependent phenomena. AIAA Journal **7**(6), 1170 – 1173 (1969)
22. Hartmann, R.: Numerical Analysis of Higher Order Discontinuous Galerkin Finite Element Methods. In: H. Deconinck (ed.) VKI LS 2008-08: CFD/ADIGMA course on very high order discretization methods. Von Karman Institute for Fluid Dynamics (2008)
23. Hughes, T.J.R., Hulbert, G.M.: Space-time finite element methods for elastodynamics: Formulations and error estimates. Computer Methods in Applied Mechanics and Engineering **66**, 339–363 (1988)
24. Hulbert, G.M., Hughes, T.J.R.: Space-time Finite Element Methods for Second-order Hyperbolic Equations. Computer Methods in Applied Mechanics and Engineering **84**(3), 327–348 (1990)
25. Jayasinghe, S., Darmofal, D.L., Galbraith, M.C., Burgess, N.K., Allmaras, S.R.: Adjoint Analysis of Buckley-Leverett and Two-phase Flow Equations. Computational Geosciences (2016 (submitted))
26. Jayasinghe, Y.S.: A Space-time Adaptive Method for Flows in Oil Reservoirs. Master's thesis, Mass. Inst. of Tech., Department of Aeronautics and Astronautics (2015)
27. Johnson, S.G.: The NLopt nonlinear-optimization package, available at: `http://ab-initio.mit.edu/nlopt`
28. Klieber, W., Rivière, B.: Adaptive simulations of two-phase flow by discontinuous galerkin methods. Computer Methods in Applied Mechanics and Engineering **196**(1-3), 404 – 419 (2006)
29. Leech, J.: Some sphere packings in higher space. Canadian Journal of Mathematics pp. 657–682 (1964)
30. Loseille, A., Alauzet, F.: Optimal 3D highly anisotropic mesh adaptation based on the continuous mesh framework. In: Proceedings of the 18th International Meshing Roundtable, pp. 575–594. Springer Berlin Heidelberg (2009)
31. Loseille, A., Löhner, R.: On 3d anisotropic local remeshing for surface, volume and boundary layers. In: Proceedings of the 18th International Meshing Roundtable, pp. 611–630. Springer Berlin Heidelberg (2009)
32. Loseille, A., Löhner, R.: Anisotropic adaptive simulations in aerodynamics. AIAA 2010-169 (2010)
33. Michel, A.: A finite volume scheme for two-phase immiscible flow in porous media. SIAM Journal on Numerical Analysis **41**, 1301–1317 (2003)
34. Obi, E., Eberle, N., Fil, A., Cao, H.: Giga Cell Compositional Simulation. International Petroleum Technology Conference (2014)
35. Oden, J.T.: A general theory of finite elements. II. applications. International Journal for Numerical Methods in Engineering **1**(3), 247–259 (1969)
36. Oden, J.T., Babuska, I., Baumann, C.E.: A discontinuous *hp* finite element method for diffusion problems. Journal of Computational Physics **146**, 491–519 (1998)
37. Peaceman, D.: Fundamentals of Numerical Reservoir Simulation. Elsevier (1977)
38. Rivière, B.: Numerical study of a discontinuous Galerkin method for incompressible two-phase flow. In: ECCOMAS Proceedings. Finland (2004)
39. Rivière, B., Wheeler, M.F.: Discontinuous Galerkin methods for flow and transport problems in porous media. Communications in Numerical Methods in Engineering **18**(1), 63–68 (2002)
40. Rivière, B., Wheeler, M.F., Banaś, K.: Part II. Discontinuous Galerkin method applied to a single phase flow in porous media. Computational Geosciences **4**, 337–349 (2000)
41. Rivière, B., Wheeler, M.F., Girault, V.: Improved energy estimates for interior penalty, constrained and discontinuous Galerkin methods for elliptic problems. Part I. Computational Geosciences **3**(3-4), 337–360 (1999)
42. Svanberg, K.: A class of globally convergent optimization methods based on conservative convex separable approximations. SIAM Journal on Optimization **12**(2), 555–573 (2002)
43. Wang, Z., Fidkowski, K., Abgrall, R., Bassi, F., Caraeni, D., Cary, A., Deconinck, H., Hartmann, R., Hillewaert, K., Huynh, H., Kroll, N., May, G., Persson, P.O., van Leer, B., Visbal, M.: High-order cfd methods: current status and perspective. International Journal for Numerical Methods in Fluids **72**(8), 811–845 (2013)

44. Wheeler, M.: An elliptic collocation-finite element method with interior penalties. SIAM Journal on Numerical Analysis **15**, 152–161 (1978)
45. Yano, M.: An optimization framework for adaptive higher-order discretizations of partial differential equations on anisotropic simplex meshes. PhD thesis, Massachusetts Institute of Technology, Department of Aeronautics and Astronautics (2012)
46. Yano, M., Darmofal, D.L.: An optimization-based framework for anisotropic simplex mesh adaptation. Journal of Computational Physics **231**(22), 7626–7649 (2012)
47. Zong, C.: Sphere Packings. Springer New York (1999)