# Multiscale gradient computation for flow in heterogeneous porous media

Jesus de Moraes, R.; Rodrigues, José R P; Hajibeygi, Hadi; Jansen, Jan Dirk

**DOI**
10.1016/j.jcp.2017.02.024

**Publication date**
2017

**Document Version**
Final published version

**Published in**
Journal of Computational Physics

Journal of Computational Physics 336 (2017) 644–663

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

# Multiscale gradient computation for flow in heterogeneous porous media

Rafael J. de Moraes [a,b,*], José R.P. Rodrigues [b], Hadi Hajibeygi [a], Jan Dirk Jansen [a]

[a] Department of Geoscience and Engineering, Faculty of Civil Engineering and Geosciences, Delft University of Technology, P.O. Box 5048, 2600, Netherlands

[b] Petrobras Research and Development Center – CENPES, Av. Horácio Macedo 950, Cidade Universitária, Rio de Janeiro, RJ 21941-915, Brazil

ARTICLE INFO

*Article history:*
Received 13 July 2016
Received in revised form 6 February 2017
Accepted 7 February 2017
Available online 14 February 2017

*Keywords:*
Gradient-based optimization
Multiscale methods
Direct method
Adjoint method
Automatic differentiation

ABSTRACT

An efficient multiscale (MS) gradient computation method for subsurface flow management and optimization is introduced. The general, algebraic framework allows for the calculation of gradients using both the Direct and Adjoint derivative methods. The framework also allows for the utilization of any MS formulation that can be algebraically expressed in terms of a restriction and a prolongation operator. This is achieved via an implicit differentiation formulation. The approach favors algorithms for multiplying the sensitivity matrix and its transpose with arbitrary vectors. This provides a flexible way of computing gradients in a form suitable for any given gradient-based optimization algorithm. No assumption w.r.t. the nature of the problem or specific optimization parameters is made. Therefore, the framework can be applied to any gradient-based study. In the implementation, extra partial derivative information required by the gradient computation is computed via automatic differentiation. A detailed utilization of the framework using the MS Finite Volume (MSFV) simulation technique is presented. Numerical experiments are performed to demonstrate the accuracy of the method compared to a fine-scale simulator. In addition, an asymptotic analysis is presented to provide an estimate of its computational complexity. The investigations show that the presented method casts an accurate and efficient MS gradient computation strategy that can be successfully utilized in next-generation reservoir management studies.



## 1. Introduction

Model-based reservoir management techniques typically rely on the information provided by derivatives. For instance, in sensitivity analysis studies, derivatives can be directly used to identify the most influential parameters in the reservoir response. Also, derivative information can be utilized in history matching [1] and control optimization [2] studies, where gradient-based optimization techniques are employed in the minimization/maximization of an objective function.

These types of model-based reservoir management studies are computationally demanding. They require multiple evaluations of the reservoir model in order to compute its response under the influence of different inputs. Reduced-order modeling (ROM) techniques have been employed to reduce the computational cost of the reservoir response evaluation. In

* Corresponding author at: Department of Geoscience and Engineering, Faculty of Civil Engineering and Geosciences, Delft University of Technology, P.O. Box 5048, 2600, Netherlands.

*E-mail addresses:* r.moraes@tudelft.nl (R.J. de Moraes), jrprodrigues@petrobras.com.br (J.R.P. Rodrigues), h.hajibeygi@tudelft.nl (H. Hajibeygi), j.d.jansen@tudelft.nl (J.D. Jansen).

http://dx.doi.org/10.1016/j.jcp.2017.02.024
0021-9991/

sensitivity analysis studies, response surface models and design of experiments are often used to reduce the computational costs (see, e.g., [3]). In history matching and optimization studies, techniques like upscaling [4], streamline simulation [5], and proper orthogonal decomposition [6] are employed to create reservoir models that are faster to evaluate. However, ROM and upscaling methods usually do not provide accurate enough system responses due to excessive simplifications of the fluid-rock physics and heterogeneous geological properties. To resolve this challenge, Multiscale (MS) methods have been developed [7–9].

MS methods solve a coarser simulation model, thus increasing the computational speed, while resolving the fine scale heterogeneities. Note that the specific multiscale methods addressed here, map between fine and coarse grids that are both at continuum (Darcy) scale, but with different computational grid resolutions. Moreover, the map between the nested fine and coarse grids is developed by using multiscale basis functions [7]. The basis functions are local solutions of the fine-scale equation, which are adaptively updated [10,11] and allow the MS coarse system to account for the fine-scale heterogeneities (which typically do not have separation of scale). Note that in contrast with MultiGrid (MG) methods, MS methods are not developed as linear solvers, but are most efficient if they are used – similar as in this paper – to provide approximate conservative fine-scale solutions (crucial for multiphase systems). MS methods are found efficient and accurate for large-scale reservoir models [12,13]. Important in this class of MS methods (compared to upscaling methods) is that the coarse-scale solutions are mapped onto the original fine scale, using the same basis functions. As such, errors can be calculated against the fine-scale reference systems (and not upscaled averaged ones). This allows for the development of convergent iterative MS procedures [14–16]. Recent developments include MS formulations for fractured media [17,18] with compositional effects [19,12,13] and complex well configurations [20] and gravitational effects [21]. In addition, algebraic formulation of the method has made it convenient to be integrated within existing simulators using structured [22,23] and unstructured [24–26] grids. The method has been also extended to fully-implicit formulations where all unknowns cross multiple dynamically-defined scales [27]. Although these developments are found efficient, they are mainly limited to forward simulation modeling. In this paper the focus is on the use of MS methods within reservoir management workflows.

Reservoir management techniques include optimization algorithms, in which calculation of derivatives plays an important role. The classical approaches for calculation of derivatives are either computationally expensive or inaccurate. For instance, numerical differentiation (see, e.g., [28,29]) suffers from discretization approximations and truncation errors, and is impractical when the number of parameters is large. Alternatively, analytical methods – Direct Methods (or Gradient Simulators) [30] and Adjoint Methods [31,32] – can provide accurate and efficient derivatives under appropriate conditions (to be further discussed in the Section 2). However, the use of analytical methods has not been extensively adopted mainly because they are code-intrusive and require a substantial implementation effort. On this issue, automatic differentiation can partly alleviate the burden of computing derivative information [33]. Additionally, most commercial simulators do not provide analytical derivative capabilities nor do they provide access to extend their functionality in this direction. Partially due to these drawbacks, ensemble methods such as the Ensemble Kalman Filter (EnKF) have become very popular in the data assimilation community [34]. Similarly, stochastic approximate gradient techniques such as ensemble optimization (EnOpt) and the stochastic simplex approximate gradient (StoSAG) method are increasingly being used for life cycle optimization [35,36]. These methods, however, by construction, provide an approximation of the gradient.

Multiscale gradient computation has been studied in the past. A Multiscale finite-volume Adjoint (MSADJ) method has been applied to sensitivity computation [37], where the global adjoint problem is solved via a set of coupled sub-grid problems described at a coarser scale. The coarse-scale sensitivities are then interpolated to the local fine grid by reconstructing the local variability of the model parameters with the aid of solving embedded adjoint sub-problems. In a follow up paper [38], the MSADJ method was efficiently applied to inverse problems of single-phase flow in heterogeneous porous media. Also, an efficient Multiscale Mixed Finite Element method has been developed for multiphase adjoint formulations, where both pressure and saturations are solved at the coarse scale [39]. In contrast to MSADJ, this method did not require fine-scale quantities.

The present development introduces a mathematical framework to compute sensitivities (gradients) in a multiscale strategy. The framework enables the same computational efficiency as existing multiscale methods [37–39]. However, its formulation allows for full flexibility with respect to the types of gradient information that are required by the different model-based reservoir management studies. This is achieved via an implicit differentiation strategy, as opposed to the more traditional Lagrange multiplier formulation. Also, the formulation naturally provides not only the Adjoint Method, but also the Direct Method. It is important to note that, although in this work the multiscale finite volume (MSFV) is being studied, the proposed MS-gradient method is not restricted to a specific MS method. Instead, it can be utilized in combination with any MS (and multi-level) strategy which is expressed in terms of restriction and prolongation operators.

This paper is structured as follows. First, the multiscale gradient computation method is derived based on the MS reservoir model equations and the respective model responses. The computation of the required prolongation (matrix of basis functions) operator derivatives is developed within the Multiscale Finite Volume (MSFV) formulation. Computational complexity of the method is also discussed from a theoretical point of view via asymptotic analysis of the algorithms. Thereafter, the Numerical Experiments section describes a systematic investigation of the validation, robustness, and accuracy of the MS-gradient method for test cases of increasing complexity. Because the proposed method is quite fundamental, the experiments are aimed at evaluating the gradient computation itself, rather than any specific application.

## 2. Derivation of the multiscale gradient computation mathematical framework

### 2.1. Forward simulation model

The derivation of the forward simulation model's response with respect to the parameters starts by firstly describing its equations in a generic, purely algebraic form. The analysis is restricted to quasi-steady state problems. In that case, the set of equations that describes the forward simulation at the fine scale can be algebraically expressed, without any assumption regarding the underlying physical model, as

$$\mathbf{g}_F(\mathbf{x}, \boldsymbol{\theta}) = \mathbf{0}, \tag{1}$$

where $\mathbf{g}_F : \mathbb{R}^{N_F} \times \mathbb{R}^{N_\theta} \rightarrow \mathbb{R}^{N_F}$ represents the set of algebraic simulator equations, $\mathbf{x} \in \mathbb{R}^{N_F}$ is the state vector (which, for single-phase flow, contains the grid block pressures), $\boldsymbol{\theta} \in \mathbb{R}^{N_\theta}$ is the vector of parameters (which typically contains grid block permeabilities and porosities but possibly also other parameters such as fault transmissibility multipliers or structural parameters), and the subscript $F$ refers to 'fine scale'. There are $N_F$ fine-scale cells and $N_\theta$ parameters. Eq. (1) implicitly assumes a dependency of the state vector $\mathbf{x}$ on the parameters $\boldsymbol{\theta}$, i.e.

$$\mathbf{x} = \mathbf{x}(\boldsymbol{\theta}). \tag{2}$$

Once the model state is determined, the observable responses of the forward model are computed. The forward model responses (typically wellbore pressures and/or rates) may not only depend on the model state, but also on the parameters themselves, and can be expressed as

$$\mathbf{y}_F = \mathbf{h}_F(\mathbf{x}, \boldsymbol{\theta}), \tag{3}$$

where $\mathbf{h}_F : \mathbb{R}^{N_F} \times \mathbb{R}^{N_\theta} \rightarrow \mathbb{R}^{N_y}$ represents the output equations [40]. It is assumed that $\mathbf{g}_F$ can be described as

$$\mathbf{g}_F(\mathbf{x}, \boldsymbol{\theta}) = \mathbf{A}(\boldsymbol{\theta})\mathbf{x} - \mathbf{q}(\boldsymbol{\theta}), \tag{4}$$

where $\mathbf{A}(\boldsymbol{\theta})$ is an $N_F \times N_F$ matrix and $\mathbf{q}(\boldsymbol{\theta})$ is an $N_F$ vector.

### 2.2. Algebraic multiscale formulation of flow in heterogeneous porous media

MS methods provide accurate and efficient solutions to the flow equations by incorporating the fine-scale heterogeneities in a coarse-scale operator [8]. This is achieved by basis functions, which are local solutions of the governing equations without right-hand-side (RHS) terms, subject to approximate local boundary conditions. These local basis functions construct the prolongation (interpolation) operator, $\mathbf{P} = \mathbf{P}(\boldsymbol{\theta})$, which is an $N_F \times N_C$ matrix that maps (interpolates) the coarse-scale solution to the fine-scale resolution. For the purpose of the development presented here, the required basic knowledge about the multiscale strategy is that a coarse-scale system can be algebraically described once the restriction operator, $\mathbf{R} = \mathbf{R}(\boldsymbol{\theta})$, is defined as an $N_C \times N_F$ matrix which maps the fine scale to the coarse scale (more information can be found in [41,22]). Let $\check{\mathbf{x}} \in \mathbb{R}^{N_C}$ be the coarse scale solution ($N_C \ll N_F$), $\mathbf{R} = \mathbf{R}(\boldsymbol{\theta})$ be an $N_C \times N_F$ matrix mapping from the fine scale to the coarse scale and $\mathbf{P} = \mathbf{P}(\boldsymbol{\theta})$ be an $N_F \times N_C$ matrix mapping from the coarse scale to the fine scale. $\check{\mathbf{x}}$ is obtained by solving

$$\check{\mathbf{g}} = (\mathbf{RAP})\check{\mathbf{x}} - (\mathbf{Rq}) = \check{\mathbf{A}}\check{\mathbf{x}} - \check{\mathbf{q}} = \check{\mathbf{0}}. \tag{5}$$

Finally, the approximated fine-scale solution $\mathbf{x}'$ is obtained by interpolating the coarse scale solution $\check{\mathbf{x}}$, i.e.,

$$\mathbf{x}' = \mathbf{P}\check{\mathbf{x}}. \tag{6}$$

### 2.3. Derivative calculation of simulator responses

In order to derive an expression to compute the multiscale derivatives with respect to the parameters, an implicit differentiation scheme is followed [42,43]. The implicit differentiation scheme facilitates both the Direct and the Adjoint methods, in contrast to the formulations based on the standard Lagrange multiplier [2,28]. In addition, through its specific algebraic form, it allows for providing any gradient information required by the selected optimization algorithm.

Based on Eq. (6), the function $\mathbf{g}'$ is defined as

$$\mathbf{g}' = \mathbf{x}' - \mathbf{P}\check{\mathbf{x}} = \mathbf{0}, \tag{7}$$

which represents the multiscale procedure to find approximate primary variables at the fine scale. The state vector is now described as a combination of both sets of primary variables at the fine and coarse scales, i.e.,

$$\mathbf{x} = \begin{bmatrix} \check{\mathbf{x}} \\ \mathbf{x}' \end{bmatrix}, \tag{8}$$

and, similarly, the model equations are represented as a combination of the equations at both scales, i.e.,

$$\mathbf{g}(\mathbf{x},\boldsymbol{\theta}) = \begin{bmatrix} \breve{\mathbf{g}} \\ \mathbf{g}' \end{bmatrix} = \mathbf{0}. \tag{9}$$

The definition of the state vector as in Eq. (8), as a key aspect of this development, allows for describing the state not only at the fine scale, but also at the coarse scale. The simulator responses $\mathbf{y}$ obtained from the multiscale method are represented as

$$\mathbf{y} = \mathbf{h}(\mathbf{x},\boldsymbol{\theta}). \tag{10}$$

The sensitivity matrix $\mathbf{G}$ is then computed by obtaining the derivative of Eq. (10) with respect to $\boldsymbol{\theta}$ as

$$\mathbf{G} = \frac{d\mathbf{h}}{d\boldsymbol{\theta}} = \frac{\partial \mathbf{h}}{\partial \mathbf{x}}\frac{d\mathbf{x}}{d\boldsymbol{\theta}} + \frac{\partial \mathbf{h}}{\partial \boldsymbol{\theta}}. \tag{11}$$

In order to find a relationship that defines the derivative of the state vector with respect to the parameters, Eq. (9) is differentiated with respect to $\boldsymbol{\theta}$

$$\frac{\partial \mathbf{g}}{\partial \mathbf{x}}\frac{d\mathbf{x}}{d\boldsymbol{\theta}} + \frac{\partial \mathbf{g}}{\partial \boldsymbol{\theta}} = \mathbf{0}, \tag{12}$$

so that

$$\frac{d\mathbf{x}}{d\boldsymbol{\theta}} = -\left(\frac{\partial \mathbf{g}}{\partial \mathbf{x}}\right)^{-1}\frac{\partial \mathbf{g}}{\partial \boldsymbol{\theta}}. \tag{13}$$

Substituting Eq. (13) in Eq. (11) gives

$$\mathbf{G} = -\frac{\partial \mathbf{h}}{\partial \mathbf{x}}\left(\frac{\partial \mathbf{g}}{\partial \mathbf{x}}\right)^{-1}\frac{\partial \mathbf{g}}{\partial \boldsymbol{\theta}} + \frac{\partial \mathbf{h}}{\partial \boldsymbol{\theta}}. \tag{14}$$

From Eqs. (5), (7), (8) and (9), it follows that

$$\frac{\partial \mathbf{g}}{\partial \mathbf{x}} = \begin{bmatrix} \mathbf{RAP} & \mathbf{0} \\ -\mathbf{P} & \mathbf{I} \end{bmatrix}. \tag{15}$$

Note that

$$\left(\frac{\partial \mathbf{g}}{\partial \mathbf{x}}\right)^{-1} = \begin{bmatrix} (\mathbf{RAP})^{-1} & \mathbf{0} \\ \mathbf{P}(\mathbf{RAP})^{-1} & \mathbf{I} \end{bmatrix} \tag{16}$$

holds. Thus, Eq. (14) can be restated as

$$\mathbf{G} = \left(\frac{\partial \mathbf{h}}{\partial \breve{\mathbf{x}}} + \frac{\partial \mathbf{h}}{\partial \mathbf{x}'}\mathbf{P}\right)(\mathbf{RAP})^{-1}\frac{\partial \breve{\mathbf{g}}}{\partial \boldsymbol{\theta}} + \frac{\partial \mathbf{h}}{\partial \mathbf{x}'}\frac{\partial \mathbf{g}'}{\partial \boldsymbol{\theta}} + \frac{\partial \mathbf{h}}{\partial \boldsymbol{\theta}}. \tag{17}$$

By deriving Eq. (5) with respect to $\boldsymbol{\theta}$ one finds

$$\frac{\partial \breve{\mathbf{g}}}{\partial \boldsymbol{\theta}} = \left[\frac{\partial \mathbf{R}}{\partial \boldsymbol{\theta}}(\mathbf{AP}) + \mathbf{R}\frac{\partial \mathbf{A}}{\partial \boldsymbol{\theta}}\mathbf{P} + (\mathbf{RA})\frac{\partial \mathbf{P}}{\partial \boldsymbol{\theta}}\right]\breve{\mathbf{x}} - \frac{\partial \mathbf{R}}{\partial \boldsymbol{\theta}}\mathbf{q} - \mathbf{R}\frac{\partial \mathbf{q}}{\partial \boldsymbol{\theta}} \tag{18}$$

and also, from Eq. (7),

$$\frac{\partial \mathbf{g}'}{\partial \boldsymbol{\theta}} = -\frac{\partial \mathbf{P}}{\partial \boldsymbol{\theta}}\breve{\mathbf{x}}. \tag{19}$$

Note that the partial derivatives of the matrices $\mathbf{A}$, $\mathbf{R}$ and $\mathbf{P}$ with respect to the vector of parameters $\boldsymbol{\theta}$ result in (third order) tensors. The appropriate interpretation of tensor operations can be found in Appendix A. The substitution of Eqs. (18) and (19) in Eq. (17) leads to an expression that will serve as the basis for the multiscale gradient computation, i.e.,

$$\begin{aligned} \mathbf{G} = {} & \left(\frac{\partial \mathbf{h}}{\partial \breve{\mathbf{x}}} + \frac{\partial \mathbf{h}}{\partial \mathbf{x}'}\mathbf{P}\right)(\mathbf{RAP})^{-1} \\ & \left\{\left[\frac{\partial \mathbf{R}}{\partial \boldsymbol{\theta}}(\mathbf{AP}) + \mathbf{R}\frac{\partial \mathbf{A}}{\partial \boldsymbol{\theta}}\mathbf{P} + (\mathbf{RA})\frac{\partial \mathbf{P}}{\partial \boldsymbol{\theta}}\right]\breve{\mathbf{x}} - \frac{\partial \mathbf{R}}{\partial \boldsymbol{\theta}}\mathbf{q} - \mathbf{R}\frac{\partial \mathbf{q}}{\partial \boldsymbol{\theta}}\right\} - \frac{\partial \mathbf{h}}{\partial \mathbf{x}'}\frac{\partial \mathbf{P}}{\partial \boldsymbol{\theta}}\breve{\mathbf{x}} + \frac{\partial \mathbf{h}}{\partial \boldsymbol{\theta}}. \end{aligned} \tag{20}$$

Eq. (20) is quite general, i.e., it can be employed to compute gradients for any MS (and multi-level) method, given that the partial derivatives of $\mathbf{R}$ and $\mathbf{P}$ are provided.

For the sake of simplicity and to be coherent with the numerical experiments presented below, from now on the dependency of the restriction operator $\mathbf{R}$ and the right-hand-side vector $\mathbf{q}$ on the parameters is neglected. This is consistent with

the MSFV method (where the restriction operator is based on a finite volume integration operator at coarse scale (see e.g. [22]). After these simplifications, Eq. (20) can be rewritten as

$$\mathbf{G} = \left(\frac{\partial \mathbf{h}}{\partial \breve{\mathbf{x}}} + \frac{\partial \mathbf{h}}{\partial \mathbf{x}'}\mathbf{P}\right)(\mathbf{RAP})^{-1}\mathbf{R}\left(\frac{\partial \mathbf{A}}{\partial \boldsymbol{\theta}}\mathbf{P} + \mathbf{A}\frac{\partial \mathbf{P}}{\partial \boldsymbol{\theta}}\right)\breve{\mathbf{x}} - \frac{\partial \mathbf{h}}{\partial \mathbf{x}'}\frac{\partial \mathbf{P}}{\partial \boldsymbol{\theta}}\breve{\mathbf{x}} + \frac{\partial \mathbf{h}}{\partial \boldsymbol{\theta}}. \tag{21}$$

The order of the operations involving the $(\mathbf{RAP})^{-1}$ term is crucial to determine the computational performance of the sensitivity matrix computation. This order defines two different algorithms known in the literature as the Direct Method and the Adjoint Method. The following sections discuss how the two methods are derived for the MS scenario defined by Eq. (21), as well as the (dis)advantages of utilizing each of them.

## 3. Computation of gradient information: generalization of the framework

According to the type of study one wants to perform, different derivative information has to be provided. For instance, for optimization methods, Quasi-Newton methods require the gradient of the objective function (and possibly constraints). In history matching algorithms, as well as in sensitivity analysis studies, the sensitivity matrix $\mathbf{G}$, defined as the matrix of derivatives of all responses with respect to all parameters, plays an important role. In Gauss–Newton methods, the matrix product $\mathbf{G}^T\mathbf{G}$ is directly used, while conjugate gradient methods require products of $\mathbf{G}$ and $\mathbf{G}^T$ with arbitrary vectors. More detailed information on the different optimization algorithms can found in [44].

To preserve the general applicability of our method, the general problem of multiplying $\mathbf{G}$ by arbitrary matrices $\mathbf{W}$ (of order $m \times N_Y$) and $\mathbf{V}$ (of order $N_\theta \times n$) from the left and the right, respectively, is considered. When $n = 1$, $\mathbf{GV}$ is simply the product of $\mathbf{G}$ with a vector, whereas, when $m = 1$, $(\mathbf{WG})^T = \mathbf{G}^T\mathbf{W}^T$ gives the product of $\mathbf{G}^T$ with the (column) vector $\mathbf{W}^T$. Those examples show how, by defining algorithms to calculate $\mathbf{GV}$ and $\mathbf{WG}$ for arbitrary $\mathbf{V}$ and $\mathbf{W}$, different types of derivative information can be accommodated in a single framework.

### 3.1. Direct method

The derivation starts by considering the calculation of $\mathbf{GV}$ for a given $N_\theta \times n$ matrix $\mathbf{V}$. From Eq. (21), by defining

$$\mathbf{Z} = \left[(\mathbf{RAP})^{-1}\mathbf{R}\left(\frac{\partial \mathbf{A}}{\partial \boldsymbol{\theta}}\mathbf{P} + \mathbf{A}\frac{\partial \mathbf{P}}{\partial \boldsymbol{\theta}}\right)\breve{\mathbf{x}}\right]\mathbf{V}, \tag{22}$$

the product $\mathbf{GV}$ can be rewritten as

$$\mathbf{GV} = \left(\frac{\partial \mathbf{h}}{\partial \breve{\mathbf{x}}} + \frac{\partial \mathbf{h}}{\partial \mathbf{x}'}\mathbf{P}\right)\mathbf{Z} - \left(\frac{\partial \mathbf{h}}{\partial \mathbf{x}'}\frac{\partial \mathbf{P}}{\partial \boldsymbol{\theta}}\breve{\mathbf{x}}\right)\mathbf{V} + \frac{\partial \mathbf{h}}{\partial \boldsymbol{\theta}}\mathbf{V}. \tag{23}$$

Here, $\mathbf{Z}$ is obtained as the solution to the following linear system of equations

$$(\mathbf{RAP})\,\mathbf{Z} = \left[\mathbf{R}\left(\frac{\partial \mathbf{A}}{\partial \boldsymbol{\theta}}\mathbf{P}\breve{\mathbf{x}} + \mathbf{A}\frac{\partial \mathbf{P}}{\partial \boldsymbol{\theta}}\breve{\mathbf{x}}\right)\right]\mathbf{V}. \tag{24}$$

This is known as the *Forward Method* [42], *Gradient Simulator* [30], or *Direct Method* [28]. Note that $\mathbf{Z}$ has dimensions of $N_C \times n$ and, therefore, it requires $n$ linear systems to be solved. Hence, the cost of computing $\mathbf{GV}$ is proportional to the number of columns in $\mathbf{V}$. In particular, to obtain the full sensitivity matrix $\mathbf{G}$ with the Direct Method, in which case one has to set $\mathbf{V}$ equal to the identity matrix of order $N_\theta$, the cost will be proportional to the number of parameters.

The algorithm to compute the product of the sensitivity matrix with a matrix via the Direct Method is depicted in Algorithm 1.

**Algorithm 1:** Right multiplying the sensitivity matrix by an arbitrary matrix via the Direct Method.

**Input** : $\mathbf{R}$, $\mathbf{A}$, $\mathbf{P}$, $\frac{\partial \mathbf{A}}{\partial \boldsymbol{\theta}}$, $\breve{\mathbf{x}}$, $\frac{\partial \mathbf{h}}{\partial \breve{\mathbf{x}}}$, $\frac{\partial \mathbf{h}}{\partial \mathbf{x}'}$, $\frac{\partial \mathbf{h}}{\partial \boldsymbol{\theta}}$, $\mathbf{V}$
**Output**: $\mathbf{GV}$

1 Compute $\boldsymbol{\alpha} = \left(\frac{\partial \mathbf{h}}{\partial \breve{\mathbf{x}}} + \frac{\partial \mathbf{h}}{\partial \mathbf{x}'}\mathbf{P}\right)$ and $\boldsymbol{\beta} = \frac{\partial \mathbf{A}}{\partial \boldsymbol{\theta}}\mathbf{P}\breve{\mathbf{x}}$
2 **foreach** $j = 1, 2, \ldots, n$ **do**
3 Compute $\boldsymbol{\gamma} = \left(\frac{\partial \mathbf{P}}{\partial \boldsymbol{\theta}}\breve{\mathbf{x}}\right)\mathbf{V}_{.,j}$ ; // Algorithm 3, where $\mathbf{m} = \mathbf{V}_{.,j}$
4 Compute $\boldsymbol{\delta} = \mathbf{R}\left(\boldsymbol{\beta}\mathbf{V}_{.,j} + \boldsymbol{A}\boldsymbol{\gamma}\right)$
5 Solve $\mathbf{z} = (\mathbf{RAP})^{-1}\boldsymbol{\delta}$
6 Compute $(\mathbf{GV})_{.,j} = \boldsymbol{\alpha}\mathbf{z} - \frac{\partial \mathbf{h}}{\partial \mathbf{x}'}\boldsymbol{\gamma} + \frac{\partial \mathbf{h}}{\partial \boldsymbol{\theta}}\mathbf{V}_{.,j}$

In Algorithm 1, the notation '., $j$' represents the $j$-th column of a matrix. Algorithm 1 requires the computation of the product of $\partial\mathbf{P}/\partial\boldsymbol{\theta}\breve{\mathbf{x}}$ by a column vector. This is discussed in the next section when Algorithm 3 is presented.

### 3.2. Adjoint method

Next the calculation of **WG** for a given $m \times N_Y$ matrix **W** is considered. From Eq. (21), defining

$$\mathbf{Z}^T = \mathbf{W}\left(\frac{\partial \mathbf{h}}{\partial \breve{\mathbf{x}}} + \frac{\partial \mathbf{h}}{\partial \mathbf{x}'}\mathbf{P}\right)(\mathbf{RAP})^{-1}, \tag{25}$$

one obtains

$$\mathbf{WG} = \mathbf{Z}^T\mathbf{R}\left(\frac{\partial \mathbf{A}}{\partial \boldsymbol{\theta}}\mathbf{P} + \mathbf{A}\frac{\partial \mathbf{P}}{\partial \boldsymbol{\theta}}\right)\breve{\mathbf{x}} - \mathbf{W}\frac{\partial \mathbf{h}}{\partial \mathbf{x}'}\frac{\partial \mathbf{P}}{\partial \boldsymbol{\theta}}\breve{\mathbf{x}} + \mathbf{W}\frac{\partial \mathbf{h}}{\partial \boldsymbol{\theta}}, \tag{26}$$

which can be rearranged as

$$\mathbf{WG} = \left(\mathbf{Z}^T\mathbf{R}\frac{\partial \mathbf{A}}{\partial \boldsymbol{\theta}}\mathbf{P}\breve{\mathbf{x}} + \left(\mathbf{Z}^T\mathbf{RA} - \mathbf{W}\frac{\partial \mathbf{h}}{\partial \mathbf{x}'}\right)\frac{\partial \mathbf{P}}{\partial \boldsymbol{\theta}}\breve{\mathbf{x}}\right) + \mathbf{W}\frac{\partial \mathbf{h}}{\partial \boldsymbol{\theta}}. \tag{27}$$

Multiplying Eq. (25) by **RAP** from the right and transposing, **Z** is obtained as the solution to the following linear system of equations

$$(\mathbf{RAP})^T\mathbf{Z} = \left(\frac{\partial \mathbf{h}}{\partial \breve{\mathbf{x}}} + \frac{\partial \mathbf{h}}{\partial \mathbf{x}'}\mathbf{P}\right)^T\mathbf{W}^T. \tag{28}$$

This is known in the literature as the *Adjoint* (or *Backward*) *Method* [32]. Note that now **Z** has dimensions of $N_C \times m$, hence it requires $m$ linear systems to be solved. As such, the cost of computing **WG** is proportional to the number of rows in **W**. In particular, to obtain the full sensitivity matrix **G** with the Adjoint Method, in which case one has to set **W** equal to the identity matrix of order $N_Y$, the cost will be proportional to the number of responses.

The algorithm to compute the product of the sensitivity matrix with a matrix from the left via the backward method is depicted in Algorithm 2.

**Algorithm 2:** Left multiplying the sensitivity matrix by an arbitrary matrix via the Adjoint Method.

**Input** : $\mathbf{R}$, $\mathbf{A}$, $\mathbf{P}$, $\frac{\partial \mathbf{A}}{\partial \boldsymbol{\theta}}$, $\breve{\mathbf{x}}$, $\frac{\partial \mathbf{h}}{\partial \breve{\mathbf{x}}}$, $\frac{\partial \mathbf{h}}{\partial \mathbf{x}'}$, $\frac{\partial \mathbf{h}}{\partial \boldsymbol{\theta}}$, $\mathbf{W}$
**Output**: $\mathbf{WG}$

1 Compute $\boldsymbol{\alpha} = \left(\frac{\partial \mathbf{h}}{\partial \breve{\mathbf{x}}} + \frac{\partial \mathbf{h}}{\partial \mathbf{x}'}\mathbf{P}\right)$ and $\boldsymbol{\beta} = \mathbf{R}\frac{\partial \mathbf{A}}{\partial \boldsymbol{\theta}}\breve{\mathbf{x}}$
2 **foreach** $i = 1, 2, ..., m$ **do**
3  Solve $\mathbf{z} = (\mathbf{RAP})^{-T}\boldsymbol{\alpha}\boldsymbol{W}_{i,.}^T$
4  Compute $\mathbf{m}^T = \left(\mathbf{z}^T\mathbf{RA} - \mathbf{W}_{i,.}\frac{\partial \mathbf{h}}{\partial \mathbf{x}'}\right)$
5  Compute $\boldsymbol{\gamma} = \mathbf{m}^T\left(\frac{\partial \mathbf{P}}{\partial \boldsymbol{\theta}}\breve{\mathbf{x}}\right)$ ; `// Algorithm 4`
6  Compute $(\mathbf{WG})_{i,.} = \mathbf{z}^T\boldsymbol{\beta} + \boldsymbol{\gamma} + \mathbf{W}_{i,.}\frac{\partial \mathbf{h}}{\partial \boldsymbol{\theta}}$

Algorithm 2 requires a left multiplication of $\partial\mathbf{P}/\partial\boldsymbol{\theta}\breve{\mathbf{x}}$ by a row vector $\mathbf{m}^T$. This will be further discussed in the next section, where Algorithm 4 is presented.

### 3.3. Remarks about the framework

This paper presents a novel technique for computation of the gradients using multiscale methods. It entails some important features which are summarized in this section.

First of all, the general formulation of the method accommodates both the Direct and Adjoint approaches. Note that the previously-developed multiscale gradient calculations were all applicable to Adjoint formulation only [37–39].

It is important to also note that the algebraic multiscale-gradient formulation, presented here, does not make any assumption with respect to neither the nature of the problem nor the specific optimization parameters. This flexible framework provides any gradient information which is required by the chosen optimization algorithm. Note that the previous methods have been developed for specific types of parameters (e.g., permeability in [37]). As such, they cannot be readily used to provide gradient information if the required parameters by the chosen optimization algorithm are not those the methods were developed for. For instance, the work presented in [37] and [38] addressed the computation of sensitivities where the parameters were specifically the grid-block permeability values. This specific parameter leads to the sensitivity of the coarse-scale quantities (transmissibility) being related to the fine-scale permeability via additional local adjoint problems for the basis functions. As such, the method presented in [37,38] is not applicable if, e.g., well controls are being used as optimization parameters. On the other hand, the work presented in [39] develops an algorithm that allows the well controls being used. However, it does not account for the sensitivity calculations of the coarse-scale quantities with respect

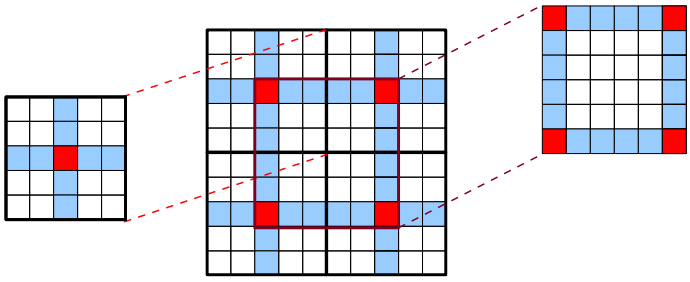

**Fig. 1.** Illustration of MSFV coarse grids for a 2D domain. Given a fine-scale grid (shown in light solid black lines), the coarse grid (shown in solid bold black) is imposed as a non-overlapping partition of the computational domain. The coarse nodes (vertices) are then selected (red cells). Connecting coarse nodes constructs the dual-coarse grid (blue cells) where basis functions are solved. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

to the fine-scale parameters. Instead, only a coarse-scale adjoint problem was solved to compute the required gradient information.

The implicit differentiation strategy of the present method makes it more flexible, compared to those methods developed based on the Lagrange multiplier formulation (as in [37–39]). Note that the Lagrange multiplier formulations are developed based on an objective function, which requires to be pre-defined for each specific application [2,28]. Instead, the presented implicit differentiation strategy [42,43] is built on a more generic system, i.e., the sensitivity matrix. More importantly, the pre- and post-multiplication of this sensitivity matrix by arbitrary matrices provides a flexible way to conform it to the specific type of gradient information needed (without requiring the pre-computation of the full sensitivity matrix).

Finally, the multiscale-gradient formulation is developed through an algebraic formulation, which is convenient to be implemented in the next-generation optimization frameworks, and also benefits all multilevel approaches that can be described through restriction and prolongation operations such as multigrid [45] and domain-decomposition [46,47] methods.

## 4. Partial derivative of MSFV prolongation operator with respect to the parameters

A particular aspect of the methodology is the computation of partial derivatives of the prolongation operator with respect to the parameters, i.e., $\partial \mathbf{P}/\partial \boldsymbol{\theta}$ in Eq. (21), and operations with this tensor. This is particularly challenging because the computation of $\mathbf{P}$ might involve complex operations, e.g., the solution of linear systems in the case of MSFV methods.

In this work, the MSFV method is considered; extensions to other MS methodologies can be obtained along the same lines. In this case, $\mathbf{P}$ is the assembly of all basis functions obtained via the solution of local flow problems, i.e.,

$$-\nabla \cdot (\lambda \cdot \nabla \varphi) = 0, \tag{29}$$

where $\lambda$ is the mobility and $\varphi$ is the basis function value. The dual-grid sub domains where the basis functions are computed are defined as follows. A primal coarse grid (on which the conservative coarse-scale system is constructed) and a dual coarse grid, which is obtained by connecting coarse nodes, are defined on a given fine-scale grid. A coarse node is a fine cell inside (typically at the center of) each coarse cell. The basis functions are solved locally on these dual coarse cells. Such overlapping coarse and dual coarse grids are crucial for conservative solutions in MSFV. An illustration of the MSFV grids is provided in Fig. 1. The prolongation operator can be expressed in terms of the basis functions corresponding to each coarse cell $j = 1, \ldots, N_C$ as

$$\mathbf{P} = \begin{bmatrix} \boldsymbol{\varphi}_1 & \boldsymbol{\varphi}_2 & \cdots & \boldsymbol{\varphi}_{N_C-1} & \boldsymbol{\varphi}_{N_C} \end{bmatrix}, \tag{30}$$

where the basis function belonging to cell $j$, $\boldsymbol{\varphi}_j$, is at column $j$, being a vector of dimension $N_F$ [22]. The construction of the basis functions $\boldsymbol{\varphi}_j$ is based on the Finite Volume (FV) discretization of Eq. (29) on the dual coarse grid cells. In order to compute the basis functions, reduced-dimensional problems are first solved at the edge cells to provide Dirichlet boundary conditions for internal cells. For the sake of clarity, let us assume a Cartesian two-dimensional solution domain (although extension to 3D is straightforward). More specifically, let $\Lambda_j$ be the set of all edges emanating from coarse node $j$ (in 2D, $\#\Lambda_j = 4$). For each edge $e \in \Lambda_j$, let

$$\mathbf{A}^E_{e,j} \boldsymbol{\varphi}^E_{e,j} - \mathbf{b}^E_{e,j} = \mathbf{0} \tag{31}$$

be the reduced-dimension discrete form of Eq. (29) along the edge $e$, where $\boldsymbol{\varphi}^E_{e,j}$ is the basis function at that edge, and $\mathbf{b}^E_{e,j}$ is the RHS which results from specifying corner values of $\varphi^E_{e,j} = 1$ at the vertex $j$ and $\varphi^E_{e,j} = 0$ at the opposite vertex of $e$. If $\Gamma_j$ is the set of all faces which have coarse node $j$ as a common vertex ($\#\Gamma_j = 4$ in 2D), for each face $f \in \Gamma_j$, according to Eq. (29), one can solve

$$\mathbf{A}^F_{f,j} \boldsymbol{\varphi}^F_{f,j} - \sum_{e \in \Lambda_j} \mathbf{E}^F_{e,f} \boldsymbol{\varphi}^E_{e,j} = \mathbf{0}, \tag{32}$$

for $\boldsymbol{\varphi}^F_{f,j}$, which is the basis function at the internal (face) cells. Also, $\mathbf{E}^F_{e,f}$ is a transformation matrix that appropriately assembles a vector represented in the edge topology of $e$ into the face topology of $f$. Note that $\mathbf{E}^F_{e,f}$ is zero when $e$ does not belong to the boundary of $f$ and that Eq. (32) implicitly defines the boundary condition as zero for any boundary cells of $f$ which do not belong to any of the edges in $\Lambda_j$. Finally, $\boldsymbol{\varphi}_j$ is the result of assembling the contribution of each $\boldsymbol{\varphi}^F_{f,j}$, $f \in \Gamma_j$, into the overall fine mesh topology:

$$\boldsymbol{\varphi}_j = \sum_{f\in\Gamma_j} \mathbf{E}_f \boldsymbol{\varphi}^F_{f,j}, \tag{33}$$

where $\mathbf{E}_f$ is a transformation matrix that assembles a vector represented in the face topology of $f$ into a size $N_F$ vector following the fine grid topology.

The derivative of $\boldsymbol{\varphi}_j$ w.r.t. the parameters is obtained from Eq. (33), i.e.,

$$\frac{\partial \boldsymbol{\varphi}_j}{\partial \boldsymbol{\theta}} = \sum_{f\in\Gamma_j} \mathbf{E}_f \frac{\partial \boldsymbol{\varphi}^F_{f,j}}{\partial \boldsymbol{\theta}}. \tag{34}$$

Differentiating Eq. (32), one can write

$$\frac{\partial \mathbf{A}^F_{f,j}}{\partial \boldsymbol{\theta}} \boldsymbol{\varphi}^F_{f,j} + \mathbf{A}^F_{f,j} \frac{\partial \boldsymbol{\varphi}^F_{f,j}}{\partial \boldsymbol{\theta}} - \sum_{e\in\Lambda_j} \mathbf{E}^F_{e,f} \frac{\partial \boldsymbol{\varphi}^E_{e,j}}{\partial \boldsymbol{\theta}} = \mathbf{0}, \tag{35}$$

from which it follows that

$$\frac{\partial \boldsymbol{\varphi}^F_{f,j}}{\partial \boldsymbol{\theta}} = \left(\mathbf{A}^F_{f,j}\right)^{-1} \left( -\frac{\partial \mathbf{A}^F_{f,j}}{\partial \boldsymbol{\theta}} \boldsymbol{\varphi}^F_{f,j} + \sum_{e\in\Lambda_j} \mathbf{E}^F_{e,f} \frac{\partial \boldsymbol{\varphi}^E_{e,j}}{\partial \boldsymbol{\theta}} \right). \tag{36}$$

Similarly, from Eq. (31),

$$\frac{\partial \boldsymbol{\varphi}^E_{e,j}}{\partial \boldsymbol{\theta}} = -\left(\mathbf{A}^E_{e,j}\right)^{-1} \frac{\partial \mathbf{A}^E_{e,j}}{\partial \boldsymbol{\theta}} \boldsymbol{\varphi}^E_{e,j} \tag{37}$$

holds. Combining Eqs. (36) and (37) into (34) results in

$$\frac{\partial \boldsymbol{\varphi}_j}{\partial \boldsymbol{\theta}} = \sum_{f\in\Gamma_j} \mathbf{E}_f \left(\mathbf{A}^F_{f,j}\right)^{-1} \left( -\frac{\partial \mathbf{A}^F_{f,j}}{\partial \boldsymbol{\theta}} \boldsymbol{\varphi}^F_{f,j} - \sum_{e\in\Lambda_j} \mathbf{E}^F_{e,f} \left(\mathbf{A}^E_{e,j}\right)^{-1} \frac{\partial \mathbf{A}^E_{e,j}}{\partial \boldsymbol{\theta}} \boldsymbol{\varphi}^E_{e,j} \right). \tag{38}$$

The third order tensor defining the derivative of $\mathbf{P}$ w.r.t. the parameters is obtained by grouping all $\partial \boldsymbol{\varphi}_j / \partial \boldsymbol{\theta}$ at its slices,

$$\frac{\partial \mathbf{P}}{\partial \boldsymbol{\theta}} = \left[ \frac{\partial \boldsymbol{\varphi}_1}{\partial \boldsymbol{\theta}} \quad \frac{\partial \boldsymbol{\varphi}_2}{\partial \boldsymbol{\theta}} \quad \cdots \quad \frac{\partial \boldsymbol{\varphi}_{N_C-1}}{\partial \boldsymbol{\theta}} \quad \frac{\partial \boldsymbol{\varphi}_{N_C}}{\partial \boldsymbol{\theta}} \right]_{N_F \times N_C \times N_\theta}. \tag{39}$$

See Appendix A for a discussion on this notation. As discussed in Algorithm 1, the product $(\partial \mathbf{P}/\partial \boldsymbol{\theta} \check{\mathbf{x}})\mathbf{m}$, where $\check{\mathbf{x}} \in \mathbb{R}^{N_C}$ and $\mathbf{m} \in \mathbb{R}^{N_\theta}$, is required. Since

$$\frac{\partial \mathbf{P}}{\partial \boldsymbol{\theta}} \check{\mathbf{x}} = \sum_{j=1}^{N_C} \frac{\partial \boldsymbol{\varphi}_j}{\partial \boldsymbol{\theta}} \check{\mathbf{x}}_j \tag{40}$$

where $\check{\mathbf{x}}_j$ denotes the $j$-th entry of vector $\check{\mathbf{x}}$, it follows, from Eq. (38), that

$$\left(\frac{\partial \mathbf{P}}{\partial} \boldsymbol{\theta} \check{\mathbf{x}}\right) \mathbf{m} = \sum_{j=1}^{N_C} \check{\mathbf{x}}_j \sum_{f\in\Gamma_j} \mathbf{E}_f \left(\mathbf{A}^F_{f,j}\right)^{-1} \left( -\left(\frac{\partial \mathbf{A}^F_{f,j}}{\partial \boldsymbol{\theta}} \boldsymbol{\varphi}^F_{f,j}\right) \mathbf{m} - \sum_{e\in\Lambda_j} \mathbf{E}^F_{e,f} \left(\mathbf{A}^E_{e,j}\right)^{-1} \left(\frac{\partial \mathbf{A}^E_{e,j}}{\partial \boldsymbol{\theta}} \boldsymbol{\varphi}^E_{e,j}\right) \mathbf{m} \right). \tag{41}$$

Algorithm 3 depicts the solution procedure to calculate $(\partial \mathbf{P}/\partial \boldsymbol{\theta} \check{\mathbf{x}})\mathbf{m}$ based on Eq. (41).

In a similar manner, Algorithm 2 requires the product $\mathbf{m}^T(\partial \mathbf{P}/\partial \boldsymbol{\theta} \check{\mathbf{x}})$, where $\check{\mathbf{x}} \in \mathbb{R}^{N_C}$ and $\mathbf{m} \in \mathbb{R}^{N_F}$. From Eqs. (38) and (40), one obtains

$$\mathbf{m}^T \left(\frac{\partial \mathbf{P}}{\partial \boldsymbol{\theta}} \check{\mathbf{x}}\right) = \sum_{j=1}^{N_C} \check{\mathbf{x}}_j \sum_{f\in\Gamma_j} \mathbf{m}^T \mathbf{E}_f \left(\mathbf{A}^F_{f,j}\right)^{-1} \left( -\frac{\partial \mathbf{A}^F_{f,j}}{\partial \boldsymbol{\theta}} \varphi^F_{f,j} - \sum_{e\in\Lambda_j} \mathbf{E}^F_{e,f} \left(\mathbf{A}^E_{e,j}\right)^{-1} \left(\frac{\partial \mathbf{A}^E_{e,j}}{\partial \boldsymbol{\theta}} \varphi^E_{e,j}\right) \right). \tag{42}$$

**Algorithm 3:** Computation of the product $(\partial \mathbf{P}/\partial\theta\check{\mathbf{x}})\mathbf{m}$, where $\check{\mathbf{x}} \in \mathbb{R}^{N_C}$ and $\mathbf{m} \in \mathbb{R}^{N_\theta}$.

**Input** : $\check{\mathbf{x}}$, $\mathbf{m}$, $\boldsymbol{\varphi}^E_{e,j}$, $\mathbf{A}^E_{e,j}$, $\frac{\partial \mathbf{A}^E_{e,j}}{\partial\boldsymbol{\theta}}$, $j = 1, ..., N_C$, $e \in \Lambda_j$, $\boldsymbol{\varphi}^F_{f,j}$, $\mathbf{A}^F_{f,j}$, $\frac{\partial \mathbf{A}^F_{f,j}}{\partial\boldsymbol{\theta}}$, $j = 1, ..., N_C$, $f \in \Gamma_j$

**Output**: $\left(\frac{\partial \mathbf{P}}{\partial\boldsymbol{\theta}}\check{\mathbf{x}}\right)\mathbf{m}$

Set: $\left(\frac{\partial \mathbf{P}}{\partial\boldsymbol{\theta}}\check{\mathbf{x}}\right)\mathbf{m} = \mathbf{0}$
**foreach** $j = 1, 2, ..., N_C$ *(primal coarse nodes)* **do**
  **foreach** $f \in \Gamma_j$ **do**
    Compute $\boldsymbol{\alpha} = -\left(\frac{\partial \mathbf{A}^F_{f,j}}{\partial\boldsymbol{\theta}}\boldsymbol{\varphi}^F_{f,j}\right)\mathbf{m}$
    **foreach** $e \in \Lambda_j$ **do**
      Compute $\boldsymbol{\beta} = \left(\frac{\partial \mathbf{A}^E_{e,j}}{\partial\boldsymbol{\theta}}\boldsymbol{\varphi}^E_{e,j}\right)\mathbf{m}$
      Solve local edge system $\boldsymbol{\gamma} = \left(\mathbf{A}^E_{e,j}\right)^{-1}\boldsymbol{\beta}$
      Accumulate result in $\boldsymbol{\alpha}$ : $\boldsymbol{\alpha} = \boldsymbol{\alpha} - \mathbf{E}^F_{e,f}\boldsymbol{\gamma}$
    Solve local face system $\boldsymbol{\delta} = \left(\mathbf{A}^F_{f,j}\right)^{-1}\boldsymbol{\alpha}$
    Accumulate result in $\left(\frac{\partial \mathbf{P}}{\partial\boldsymbol{\theta}}\check{\mathbf{x}}\right)\mathbf{m}$ : $\left(\frac{\partial \mathbf{P}}{\partial\boldsymbol{\theta}}\check{\mathbf{x}}\right)\mathbf{m} = \left(\frac{\partial \mathbf{P}}{\partial\boldsymbol{\theta}}\check{\mathbf{x}}\right)\mathbf{m} + \check{\mathbf{x}}_j\mathbf{E}_f\boldsymbol{\delta}$

By defining

$$\boldsymbol{\gamma}^T_{f,j} = \mathbf{m}^T\mathbf{E}_f\left(\mathbf{A}^F_{f,j}\right)^{-1} \tag{43}$$

and

$$\boldsymbol{\delta}^T_{e,f,j} = \gamma^T\mathbf{E}^F_{e,f}\left(\mathbf{A}^E_{e,j}\right)^{-1}, \tag{44}$$

Eq. (42) can be rewritten as

$$\mathbf{m}^T\left(\frac{\partial \mathbf{P}}{\partial\boldsymbol{\theta}}\check{\mathbf{x}}\right) = \sum_{j=1}^{N_C}\check{\mathbf{x}}_j\sum_{f\in\Gamma_j}\left(-\boldsymbol{\gamma}^T_{f,j}\left(\frac{\partial \mathbf{A}^F_{f,j}}{\partial\boldsymbol{\theta}}\boldsymbol{\varphi}^F_{f,j}\right) - \sum_{e\in\Lambda_j}\boldsymbol{\delta}^T_{e,f,j}\left(\frac{\partial \mathbf{A}^E_{e,j}}{\partial\boldsymbol{\theta}}\varphi^E_{e,j}\right)\right). \tag{45}$$

Right multiplying Eq. (43) by $\mathbf{A}^F_{f,j}$ and transposing, one can see that $\boldsymbol{\gamma}_{f,j}$ is obtained as

$$\left(\mathbf{A}^F_{f,j}\right)^T\boldsymbol{\gamma}_{f,j} = \mathbf{E}^T_f\mathbf{m}. \tag{46}$$

Note that $\mathbf{E}^T_f$ is the transformation matrix that assembles a vector following the fine grid topology into its proper restriction to the face topology of $f$. Analogously, $\boldsymbol{\delta}_{e,f,j}$ is obtained by solving

$$\left(\mathbf{A}^E_{e,j}\right)^T\boldsymbol{\delta}_{e,f,j} = \left(\mathbf{E}^F_{e,f}\right)^T\boldsymbol{\gamma}. \tag{47}$$

Similar as for $\mathbf{E}^T_f$, $(\mathbf{E}^F_{e,f})^T$ is the transformation matrix that assembles a vector following the face topology of $f$ into its restriction to the edge topology of $e$. Equations (45), (46) and (47) are the basis for the algorithm to calculate $\mathbf{m}^T(\partial\mathbf{P}/\partial\boldsymbol{\theta}\check{\mathbf{x}})$, as depicted in Algorithm 4. Note that the face systems are solved before the edge ones, in contrast to the order of the MS calculation.

### 4.1. Prolongation and its derivative in the presence of wells

Wells and other fine-scale source terms are considered in the multiscale formulation by supplementary well basis functions [20,17]. Well functions are local solutions to Eq. (29) subject to Dirichlet conditions of 1 at the well cell and 0 at the coarse nodes. Hence, considering well functions, the prolongation operator (for porous rock) is enriched and reads

$$\mathbf{P} = \left[\begin{array}{ccc|ccc}\boldsymbol{\varphi}_1 & \cdots & \boldsymbol{\varphi}_{N_C} & \psi_1 & \cdots & \psi_{N_W}\end{array}\right], \tag{48}$$

where each well function $\boldsymbol{\psi}_w$ adds a column vector to the porous rock prolongation operator. Note that there are $N_W$ wells in the domain. For rate-constrained wells, one has to add additional rows to the prolongation (and consequently, the coarse-scale system) in order to solve for the well pressures as additional unknowns [17]. The derivative of (48) w.r.t. the parameters is given by

**Algorithm 4:** Computation of the product $\mathbf{m}^T(\partial\mathbf{P}/\partial\theta\check{\mathbf{x}})$, where $\check{\mathbf{x}} \in \mathbb{R}^{N_C}$ and $\mathbf{m} \in \mathbb{R}^{N_F}$.

**Input** : $\check{\mathbf{x}}$, $\mathbf{m}$, $\varphi^E_{e,j}$, $\mathbf{A}^E_{e,j}$, $\frac{\partial\mathbf{A}^E_{e,j}}{\partial\boldsymbol{\theta}}$, $j = 1, ..., N_C$, $e \in \Lambda_j$, $\varphi^F_{f,j}$, $\mathbf{A}^F_{f,j}$, $\frac{\partial\mathbf{A}^F_{f,j}}{\partial\boldsymbol{\theta}}$, $j = 1, ..., N_C$, $f \in \Gamma_j$

**Output**: $\mathbf{m}^T\left(\frac{\partial\mathbf{P}}{\partial\boldsymbol{\theta}}\check{\mathbf{x}}\right)$

Set: $\mathbf{m}^T\left(\frac{\partial\mathbf{P}}{\partial\boldsymbol{\theta}}\check{\mathbf{x}}\right) = 0$
**foreach** $j = 1, 2, ..., N_C$ **do**
  **foreach** $f \in \Gamma_j$ **do**
    Solve transpose local face system $\gamma = \left(\mathbf{A}^F_{f,j}\right)^{-T}\mathbf{E}^T_f\mathbf{m}$
    Compute $\alpha = -\gamma^T\left(\frac{\partial\mathbf{A}^F_{f,i}}{\partial\boldsymbol{\theta}}\varphi^F_{f,i}\right)$
    **foreach** $e \in \Lambda_j$ **do**
      Solve transpose local edge system $\delta = \left(\mathbf{A}^E_{e,j}\right)^{-T}\left(\mathbf{E}^F_{e,f}\right)^T\gamma$
      Accumulate result in $\alpha$ : $\alpha = \alpha - \delta^T\left(\frac{\partial\mathbf{A}^E_{e,j}}{\partial\boldsymbol{\theta}}\varphi^E_{e,j}\right)$
    Accumulate result in $\mathbf{m}^T\left(\frac{\partial\mathbf{P}}{\partial\boldsymbol{\theta}}\check{\mathbf{x}}\right)$ : $\mathbf{m}^T\left(\frac{\partial\mathbf{P}}{\partial\boldsymbol{\theta}}\check{\mathbf{x}}\right) = \mathbf{m}^T\left(\frac{\partial\mathbf{P}}{\partial\boldsymbol{\theta}}\check{\mathbf{x}}\right) + \check{\mathbf{x}}_j\alpha$

$$\frac{\partial\mathbf{P}}{\partial\boldsymbol{\theta}} = \left[\begin{array}{ccc|ccc} \frac{\partial\varphi_1}{\partial\boldsymbol{\theta}} & \cdots & \frac{\partial\varphi_{N_C}}{\partial\boldsymbol{\theta}} & \frac{\partial\boldsymbol{\psi}_1}{\partial\boldsymbol{\theta}} & \cdots & \frac{\partial\boldsymbol{\psi}_{N_W}}{\partial\boldsymbol{\theta}} \end{array}\right]. \tag{49}$$

Computation of the partial derivatives of the well basis functions follows similar strategy as discussed for Eq. (38).

## 5. Computational aspects of the MS-gradient method

### 5.1. Partial derivative computation and automatic differentiation

In the computation of analytical gradients, a significant part of the overall implementation effort is associated with the computation of partial derivative matrices. This computation step often requires access to the source code because not all partial derivatives are readily available, as opposed to the state variable derivatives via the Jacobian matrix (see, e.g., inputs required by Algorithm 1 and Algorithm 2). To calculate the partial derivative matrices an Automatic Differentiation (AD) approach is used, because of its flexibility and accuracy. A review of different AD approaches is provided by [33]. In our work, an Operator Overloading AD technique (Bendtsen and Stauning 1996) based on Expression Templates [48] is applied. This facilitates the computation and assemblage of all the partial derivative matrices because they are obtained at the same time as when the system matrix and the simulator response are computed.

### 5.2. Computational efficiency

An asymptotic analysis is performed to assess the efficiency of the MS-gradient information, compared with the fine-scale gradient computation. Note that the computational cost of linear system assembly and matrix-vector products are negligible when compared to the cost involved in solving the linear system of equations. The complexity of solving a linear system of size $N$ is assumed to be $\mathcal{O}(aN^b)$, where $a$ and $b$ are constants associated with the specific linear solver employed.

Following Algorithm 1, $n$ (number of columns in the $\mathbf{V}$ matrix) linear systems of size $N_C$ must be solved to fully define the gradient information, hence its computational complexity reads $\mathcal{O}(n(a_{MS}{N_C}^{b_{MS}}))$. Additionally, Algorithm 3 consists of solving $N_L \times n \times N_C$ linear systems of size $N_R = N_F/N_C$ to provide the partial derivative of $\mathbf{P}$ w.r.t. the parameters. Here, $N_R$ is the multiscale coarsening ratio, and $N_L$ is the number of local problems that must be solved per coarse grid vertex (4 in 2D and 8 in 3D problems). Thus, the computational complexity of Algorithm 3 is $\mathcal{O}(N_L n N_C(a_{MS}{N_R}^{b_{MS}}))$, which results in a complexity of $\mathcal{O}^D_{MS}(n(a_{MS}{N_C}^{b_{MS}} + N_L N_C(a_{MS}{N_R}^{b_{MS}})))$ for the MS Direct Method.

A similar analysis can be performed for Algorithm 2 and Algorithm 4, resulting in an estimate of the complexity of the MS Adjoint Method, i.e., $\mathcal{O}^A_{MS}(m(a_{MS}{N_C}^{b_{MS}} + N_L N_C(a_{MS}{N_R}^{b_{MS}})))$, where $m$ is the number of columns of $\mathbf{W}$.

The complexity of the fine-scale gradient computation for the Direct and Adjoint Methods is given by $\mathcal{O}(n(a_{FS}{N_C}^{b_{FS}}))$ and $\mathcal{O}(m(a_{FS}{N_C}^{b_{FS}}))$, respectively. Therefore, the cost ratio between the MS and fine-scale computational efficiency can be defined as

$$\frac{\mathcal{O}_{MS}}{\mathcal{O}_{FS}} = \frac{a_{MS}}{a_{FS}}\left(\frac{{N_C}^{b_{MS}}}{{N_F}^{b_{FS}}} + N_L\frac{{N_F}^{b_{MS}}}{{N_F}^{b_{FS}}}{N_C}^{(1-b_{MS})}\right), \tag{50}$$

which holds for both the Direct and the Adjoint Method.

For the sake of simplicity, it is assumed that the solver employed to the MS system is equally efficient to the one employed to the fine-scale system, i.e., $a_{MS} = a_{FS}$ and $b_{MS} = b_{FS} = b$. As such, the expression defining the cost ratio simplifies to

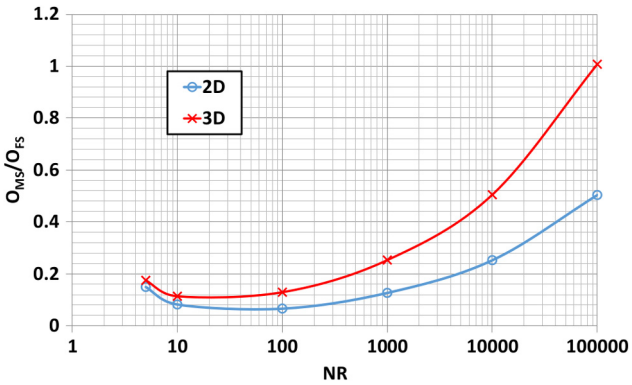


**Fig. 2.** Cost ratio between MS and fine-scale gradient computation methods, as a function of the multiscale coarsening ratio, $N_R$, for a 2D (blue) and 3D (red) domain with $N_F = 10^7$ fine-scale cells. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

$$\frac{\mathcal{O}_{MS}}{\mathcal{O}_{FS}} = \frac{1}{N_R^b} + N_L \frac{N_R^{b-1}}{N_F^{b-1}}. \tag{51}$$

For a fixed number of fine grid cells $N_F$, it is mainly the coarsening ratio $N_R$ that determines the complexity of the MS-gradient algorithm. To illustrate this point, a domain with $N_F = 10^7$ fine-scale grid cells is considered. The MS-gradient speed-up, $\mathcal{O}_{MS}/\mathcal{O}_{FS}$, for different coarsening ratios $N_R$ for both 2D and 3D cases is presented in Fig. 2, where $b = 1.3$.

## 6. Numerical experiments

In this section, performance of the MS-gradient method is studied for single-phase incompressible flow in heterogeneous porous media. The following numerical experiments are presented to first validate and then assess the accuracy of the gradient information computed by the method. For this purpose, a misfit objective function with no regularization term

$$O(\boldsymbol{\theta}) = \frac{1}{2}(\mathbf{h}(\mathbf{x},\boldsymbol{\theta}) - \mathbf{d}_{obs})^T \mathbf{C}_D^{-1} (\mathbf{h}(\mathbf{x},\boldsymbol{\theta}) - \mathbf{d}_{obs}), \tag{52}$$

with a gradient

$$\nabla_\theta O = \mathbf{G}^T \mathbf{C}_D^{-1} (\mathbf{h}(\mathbf{x},\boldsymbol{\theta}) - \mathbf{d}_{obs}), \tag{53}$$

is considered [28]. In all experiments, the fitting parameters are cell-centered permeabilities. The observed quantity, $\mathbf{d}_{obs}$, is the fine scale pressure at the location of (non-flowing) observation wells, therefore

$$\frac{\partial \mathbf{h}}{\partial \mathbf{x}'} = \mathbf{I}, \tag{54}$$

and

$$\frac{\partial \mathbf{h}}{\partial \check{\mathbf{x}}} = \mathbf{0}. \tag{55}$$

The tensor $\partial\mathbf{A}/\partial\boldsymbol{\theta}$ represents partial derivatives of the system (transmissibility) matrix with respect to permeability. Note that the results are expressed in terms of a non-dimensional pressure, i.e.,

$$p_D = \frac{p - p_{prod}}{p_{inj} - p_{prod}}, \tag{56}$$

where $p_{inj}$ and $p_{prod}$ are the injection and production pressures, respectively. In all the experiments, $p_{inj} = 1.0$ and $p_{prod} = 0.0$, the grid-block dimensions are $\Delta x = \Delta y = \Delta z = 1$ m and the fluid viscosity is $1.0 \times 10^{-3}$ Pa s. In addition, in all the following test cases, well basis functions are included.

### 6.1. Validation experiments

The MS-gradient method is validated against the numerical differentiation method with a higher-order, two-sided Taylor approximation

$$\nabla_{\boldsymbol{\theta}} O_i = \frac{1}{2\delta\theta_i}\left(O(\theta_i,\ldots,\theta_{i-1},\theta_i+\delta\theta_i,\theta_{i+i},\ldots,\theta_{N_\theta}) - O(\theta_i,\ldots,\theta_{i-1},\theta_i-\delta\theta_i,\theta_{i+i},\ldots,\theta_{N_\theta})\right), \tag{57}$$

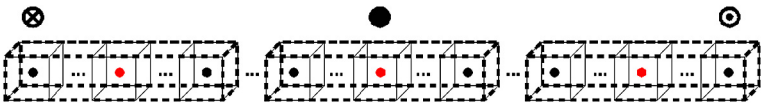

**Fig. 3.** Fine and coarse grids and wells setup for 1D numerical experiments. The solid thin lines represent the fine grid-blocks. The bold dashed lines represent the primal-coarse grid blocks. Vertex cells identified with red circles. The crossed circle represents the injection well, the dotted circle the production well, and the solid circle the observation well. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

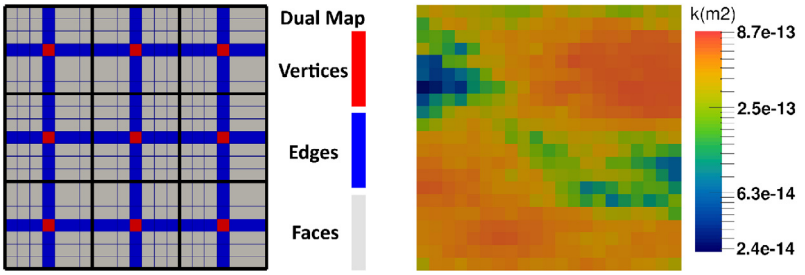


**Fig. 4.** (a) Fine, primal and dual coarse grids for 2D validation test cases and (b) reference permeability field.

where $\delta$ is a multiplicative parameter perturbation. The relative error can be defined as

$$\varepsilon = \frac{\|\nabla_{\boldsymbol{\theta}} O_{FD} - \nabla_{\boldsymbol{\theta}} O_{AN}\|_2}{\|\nabla_{\boldsymbol{\theta}} O_{AN}\|_2}, \tag{58}$$

where $\nabla_{\boldsymbol{\theta}} O_{FD}$ is obtained by performing the appropriate number of multiscale reservoir simulations required to evaluate Eq. (57) and $\nabla_{\theta} O_{AN}$ is obtained by either employing the Direct or the Adjoint Method to evaluate Eq. (53). Note that

$$\mathbf{m} = \mathbf{C}_D^{-1} \left(\mathbf{h}\left(\mathbf{x}, \boldsymbol{\theta}\right) - \mathbf{d}_{obs}\right), \tag{59}$$

where $\mathbf{m}$ is an auxiliary vector, so the gradient of $O$ can be written as $\nabla_{\boldsymbol{\theta}} O = (\mathbf{m}^T \mathbf{G})^T$ and Algorithm 1, with $\mathbf{W} = \mathbf{m}^T$, calculates $\nabla_{\boldsymbol{\theta}} O$ with a cost proportional to one extra MS simulation, making the Adjoint Method a much more efficient way to calculate the gradient than the Direct Method. Moreover, for all cases, unless stated otherwise, for simplicity, it is assumed $\mathbf{C}_D = \mathbf{I}$.

In order to validate the proposed derivative calculation methods, as well as their implementation, the linear decrease of the error $\varepsilon$ by decreasing the perturbation value $\delta$ (see chapter 8 of [29]) from $10^{-1}$ to $10^{-4}$ (the range within which only discretization errors are observed) is investigated.

The investigation is carried out in three examples of increasing complexity. The first case is a one-dimensional, homogeneous medium with 45 grid blocks. A primal coarse grid of just 3 grid blocks is employed (coarsening ratio of 15). Injection and production wells are located at, respectively, cells 1 and 45. One observation well is located at the center of the domain, i.e. at grid block 23. The pressure measured in the observation well is taken from a reference case with a randomly sampled permeability field. The non-dimensional injection and production pressures are one and zero, respectively. Fig. 3 illustrates the setup for this experiment.

In the other two experiments, the accuracy of the method is assessed for 2D test cases, one homogeneous and another one heterogeneous. In both cases the fine grid size is $21 \times 21$, while the coarse grid size is $3 \times 3$ (coarsening ratio of $7 \times 7$). The primal- and dual-coarse grids are illustrated in Fig. 4(a).

The permeability field is extracted from 1000 geological realizations, as shown in Fig. 4(b), and serves to compute the observed pressure. For the heterogeneous case, another geological realization is chosen from the ensemble. The ensemble is generated via the decomposition of a reference permeability "image" using Principal Component Analysis parameterization. Fig. 5 illustrates 4 different permeability realizations from the ensemble. See [49] for more details.

A quarter five-spot well configuration is considered, with two observation wells close to the operating wells. The well positions and operating pressures are described in Table 1.

Fig. 6 shows that the MS-gradient and fine-scale methods have the same order of accuracy with respect to the perturbation $\delta$, for all cases. The expected behavior of linearly decreasing error values as the perturbation size decreases is observed in all experiments. One can also note that the Direct and Adjoint Methods are equally accurate, i.e., they both provide the analytical gradient at the same accuracy. Also, it is important to notice that the localization assumptions involved in 2D are consistently represented by the analytical methods. The result of the third experiment (Fig. 6(c)) indicates the correctness of the method when applied to compute gradients for heterogeneous media.

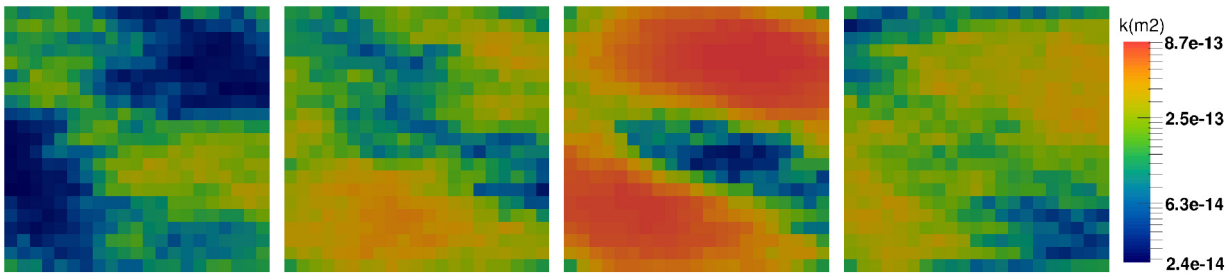


**Fig. 5.** Four different permeability realizations from the ensemble of 1000 members used in the 2D numerical experiments.

**Table 1**
Well configuration for the homogeneous, two-dimensional case.

| Well | Fine scale position (I, J) | Well type |
|---|---|---|
| INJE | (1, 1) | Injection |
| PROD | (21, 21) | Production |
| OBSWELL1 | (3, 3) | Observation |
| OBSWELL2 | (19, 19) | Observation |

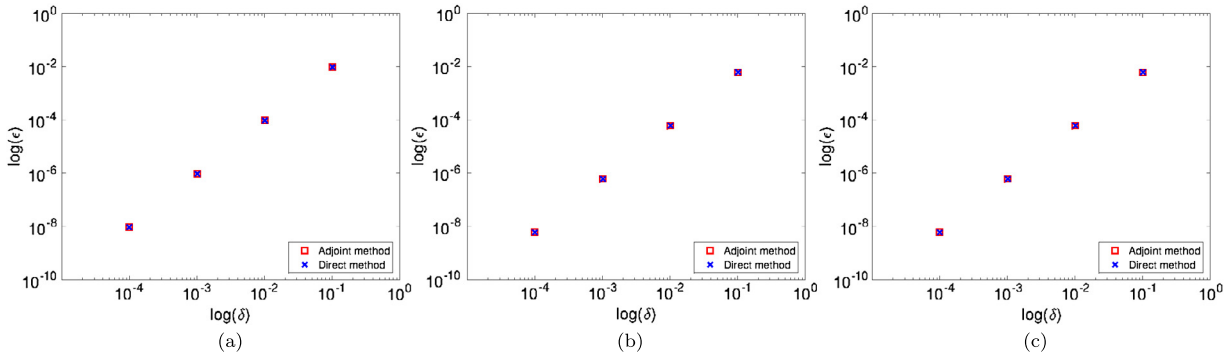


**Fig. 6.** Validation of MS gradient computation method via comparison with numerical differentiation. (a) One-dimensional, homogeneous, (b) two-dimensional homogeneous and (c) two-dimensional heterogeneous steady-state flow test cases.

### 6.2. Gradient accuracy

In order to assess the quality of the MS gradient, the angle between fine-scale and MS normalized gradients, i.e.,

$$\alpha = \cos^{-1}\left(\nabla_{\mathbf{\theta}}^{T}\,\hat{O}_{FS}\,\nabla_{\mathbf{\theta}}\hat{O}_{MS}\right), \tag{60}$$

is measured. Here,

$$\nabla_{\mathbf{\theta}}\hat{O}_{FS} = \frac{\nabla_{\mathbf{\theta}}O_{FS}}{\|\nabla_{\mathbf{\theta}}O_{FS}\|_2} \tag{61}$$

and

$$\nabla_{\mathbf{\theta}}\hat{O}_{MS} = \frac{\nabla_{\mathbf{\theta}}O_{MS}}{\|\nabla_{\mathbf{\theta}}O_{MS}\|_2}. \tag{62}$$

Also, $\nabla_{\mathbf{\theta}}O_{FS}$ and $\nabla_{\mathbf{\theta}}O_{MS}$ denote the fine-scale and MS analytical gradients, respectively. As a minimum requirement, acceptable MS gradients are obtained if $\alpha$ is much smaller than 90° [50].

Firstly, for the 1D test case, both methods result in $\alpha = 0°$, indicating that fine scale and MS gradients are perfectly aligned in this case. This is due to the fact that, in 1D, no approximations (due to localization) are made in the MS solution, and thus, in the MS gradient computation.

For the 2D homogeneous case, the fine scale and MS gradients result in $\alpha = 10.97°$, using the same setup depicted in Fig. 4(a). Although the gradients are practically pointing in the same direction, a deviation between the two is observed. To better investigate this difference, Fig. 7(a) and Fig. 7(b) present the fine-scale and MS pressure solutions, respectively, and Fig. 7(c) shows the difference between these two pressure solutions. Fig. 7(d) and Fig. 7(e) present the absolute, normalized

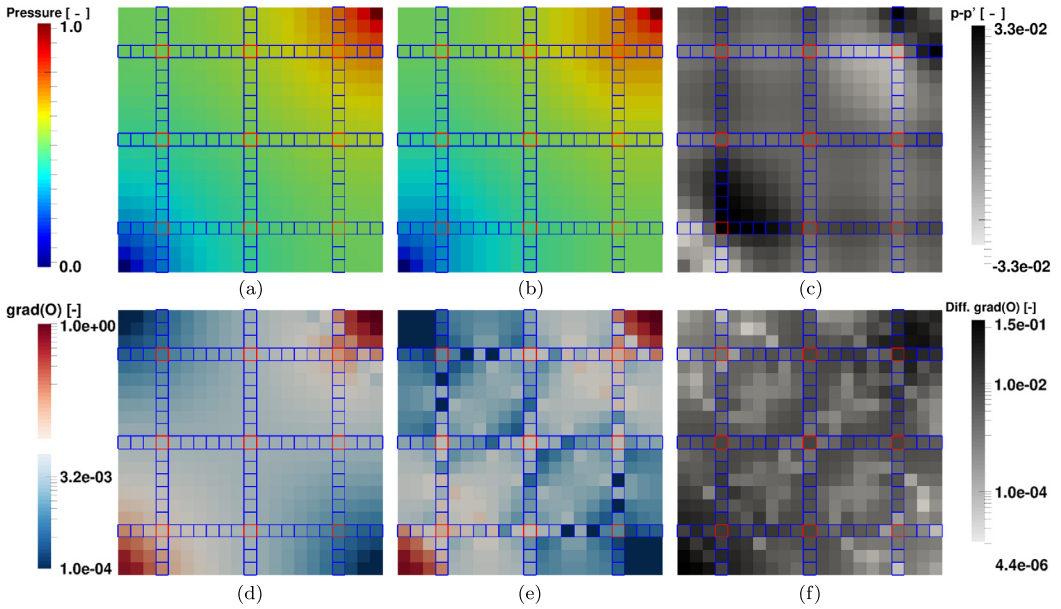


**Fig. 7.** Fine scale (p) pressure solution (a) and MSFV ($p'$) pressure solution (b). Difference between fine-scale and MSFV pressure solutions (c). Absolute, normalized fine scale gradient (d) and MSFV (e). Difference between fine-scale and MS absolute, normalized gradients (f). In all figures the dual-coarse grid is shown. Also shown, in the red boxes, are the coarse nodes. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

**Table 2**
Well configuration for the homogeneous, two-dimensional case.

| Well | Fine scale position (I, J) | Well type |
|---|---|---|
| INJE | (11, 11) | Injection |
| PROD1 | (1, 1) | Production |
| PROD2 | (21, 1) | Production |
| PROD3 | (1, 21) | Production |
| PROD4 | (21, 21) | Production |
| OBSWELL1 | (3, 3) | Observation |
| OBSWELL2 | (19, 3) | Observation |
| OBSWELL3 | (3, 19) | Observation |
| OBSWELL4 | (19, 19) | Observation |

gradient of the OF computed via fine-scale and MS gradient methods, respectively. Fig. 7(f) shows the difference between the absolute, normalized gradient computed by the two methods.

The difference between the MS and fine-scale methods is due to the localization assumption in the calculation of the basis functions [14]. Note that, thanks to the well functions, the multiscale solutions are accurately capturing the wells.

### 6.3. *Effect of heterogeneity distribution and coarsening ratio*

In the first case, OF gradients are computed for the whole ensemble of heterogeneous permeability fields (see Fig. 4). Moreover, the same realization depicted in Fig. 4(b) is considered as the reference from which the observed pressures are computed.

An inverted five-spot well pattern is employed, whiled four observation wells are placed close to the production wells. The well configuration is depicted in Table 2.

Two different coarsening ratios are applied, one resulting in a coarse grid of $3 \times 3$ (as illustrated in Fig. 4(a)) and another one resulting in a coarse grid of $7 \times 7$. The angles between the fine scale gradient and the MS gradient for each realization are computed, using Eq. (60). A histogram illustrating the angle distribution for each coarsening ratio is presented in Fig. 8.

Note that, for this case, the quality of the gradients is significantly improved once the coarse grid size of $3 \times 3$ is increased to $7 \times 7$. Important to note is that, as shown in Fig. 9, the MS gradients are least accurate when the norm of the gradient vector is small. Therefore, they are not expected to have a major effect on the optimization procedure.

For the $3 \times 3$ coarse grid case, 9.71% of the angles are greater than 90°, indicating that some MS gradients point in the opposite direction of the decreasing OF direction. On the other hand, if a $7 \times 7$ coarse grid is used no angle is greater than 90°.

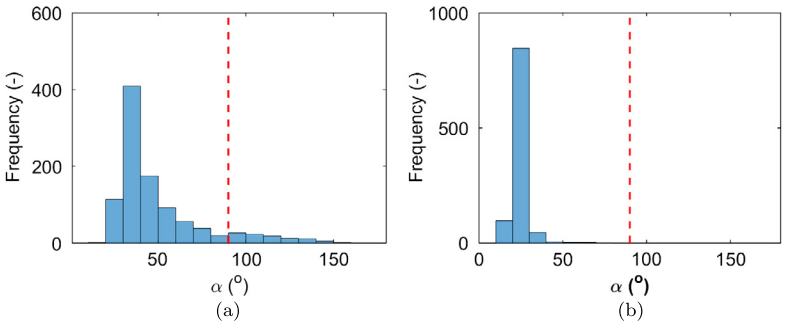


**Fig. 8.** Histograms of angles between fine scale and multiscale gradients for $3 \times 3$ (a) and $7 \times 7$ (b) coarse grids. The fine-scale computational domain contains $21 \times 21$ grid cells. The vertical red dashed line indicates the 90° limit. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

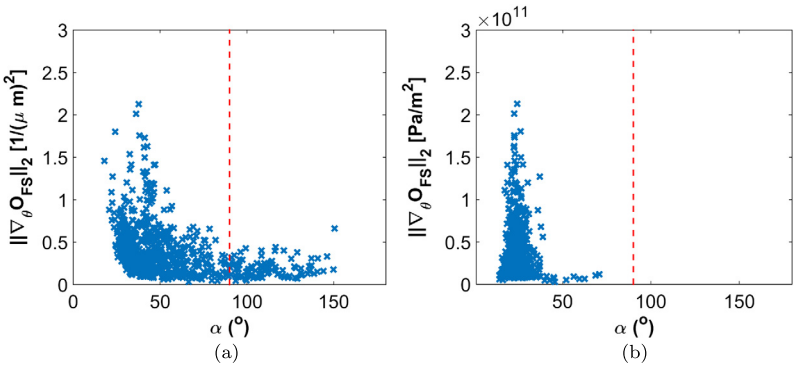


**Fig. 9.** Cross-plots of fine scale gradient norm vs. $\alpha$ for the 1000 realizations in the heterogeneous ensemble for (a) $3 \times 3$ and (b) $7 \times 7$ coarse grid. Note that MS gradients are accurate for the cases with large values of gradient norms. Their relatively inaccurate estimates (large $\alpha$) happen mostly when the norms of gradients are small. The vertical red dashed line indicates the 90° limit. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

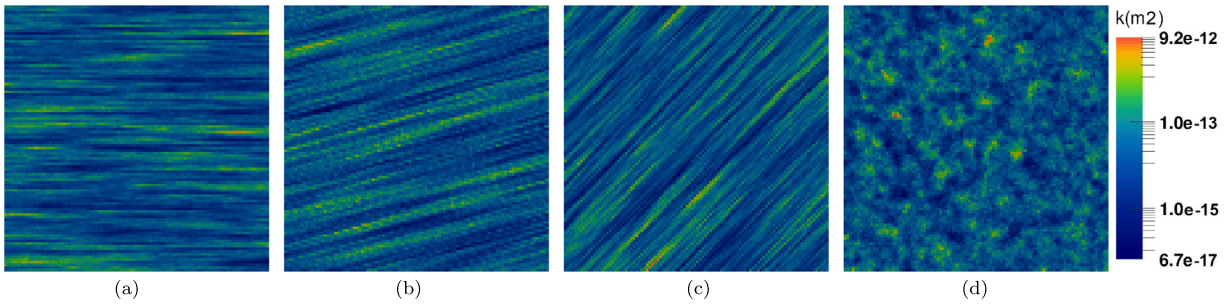


**Fig. 10.** Permeability distribution of four different realizations taken from the sets of 20 geostatistically equiprobable permeability fields with 0° (a), 15° (b), and 45° (c) correlation angles. Also, a patchy field (d) with a small correlation length is considered.

In order to further explore this point, four sets of 20 equiprobable realizations of log-normally distributed permeability fields with a spherical variogram and dimensionless correlation lengths of $\Psi_1 = 0.5$ and $\Psi_2 = 0.02$ are generated using sequential Gaussian simulations [51]. For each set, the variance and the mean of $\ln(k)$ are 2.0 and 3.0, respectively, where $k$ is the grid block permeability. As depicted in Fig. 10, for the realizations with a long correlation length, the angles between the permeability layers and the horizontal axis are 0°, 15°, and 45°. A patchy (small correlation length) pattern is also considered (Fig. 10(d)). Compared with the previous set, the permeability contrast is much higher in this case.

The fine-scale and coarse grids contain $100 \times 100$ and $20 \times 20$ cells, respectively. The well configuration utilized in this numerical experiment is depicted in Table 3.

From Fig. 11, one can observe that all cases with $\alpha > 50^o$ are associated with small gradient norms and that, for all geological sets, the MS gradient provides gradient directions that are very accurate, compared with the fine scale solution, when the gradient norms are large.

Note also that, in the case of a small gradient norm, the OF has a weak dependency on the parameters in the vicinity of the point where the gradient is being calculated. As such, the overall performance of the optimization algorithm is not

**Table 3**
Well configuration for Case 2.

| Well | Fine scale position (I, J) | Well type |
|---|---|---|
| INJE | (1, 1) | Injection |
| PROD | (100, 100) | Production |
| OBSWELL1 | (3, 3) | Observation |
| OBSWELL2 | (98, 98) | Observation |

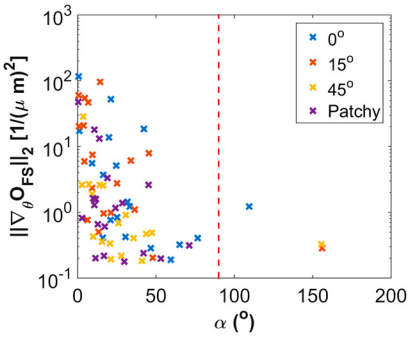


**Fig. 11.** Cross-plot of fine scale gradient norm (log scale) vs. $\alpha$ for the four sets of 20 equiprobable permeability realizations. The vertical red dashed line indicates the 90° limit. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

expected to be affected by replacing the fine-scale gradient calculation with the MS version. For practical purposes, as will be shown by the next numerical experiment, acceptable results in optimization studies can be achieved by using approximated multiscale gradients. In general, an iterative multiscale strategy [14] should be used in order to guarantee the quality of the multiscale gradient. The key component of such a development is to relate the quality of the approximate parameters (often measured via residuals) with the quality of the gradients. The employment of an iterative scheme imposes challenges associated with, for instance, the computation of extra partial derivative information that arises from the smoothing step. These challenges fall beyond the scope of this paper.

### 6.4. Parameter estimation study

In order to investigate how the approximate MS gradient performs in an optimization algorithm, a parameter estimation study is performed. In this study, the same permeability ensemble as illustrated in Fig. 5 is used. The permeability field employed to create the synthetic data is illustrated in Fig. 4(b). The initial guess is randomly chosen from the ensemble. The MS method employs $7 \times 7$ coarse grid cells. The well configuration is presented in 1. Differently from the other experiments, a misfit objective function with a regularization term [28], i.e.

$$O\left(\mathbf{y}, \boldsymbol{\theta}\right) = \frac{1}{2}\left(\boldsymbol{\theta} - \boldsymbol{\theta}_{prior}\right)^T \mathbf{C}_\theta^{-1}\left(\boldsymbol{\theta} - \boldsymbol{\theta}_{prior}\right) + \frac{1}{2}\left(\mathbf{h}\left(\mathbf{x}, \boldsymbol{\theta}\right) - \mathbf{d}_{obs}\right)^T \mathbf{C}_D^{-1}\left(\mathbf{h}\left(\mathbf{x}, \boldsymbol{\theta}\right) - \mathbf{d}_{obs}\right), \tag{63}$$

is considered, where $\theta \in \mathbb{R}^{N_F}$ is the vector of parameters, taken to be the natural logarithm of the permeability in each grid cell. The covariance matrix $\mathbf{C}_\theta$ is computed from the ensemble of realizations as

$$\mathbf{C}_\theta = \frac{1}{N_e - 1}\left(\boldsymbol{\Theta} - \boldsymbol{\mu}\mathbf{e}^T\right)\left(\boldsymbol{\Theta} - \boldsymbol{\mu}\mathbf{e}^T\right)^T \tag{64}$$

where $\boldsymbol{\Theta}$ is the $N_F \times N_e$ matrix whose $j$-th column is given by the member of the ensemble $\theta_j$, $j \in \{1, ..., N_e\}$,

$$\mu = \frac{1}{N_e}\sum_{j=1}^{N_e} \boldsymbol{\theta}_j \tag{65}$$

is the ensemble mean, and $\mathbf{e} = [\mathbf{1}, ..., \mathbf{1}]^T$ is a vector of ones of size $N_e \times 1$. In Eq. (63), the prior is taken to be the ensemble mean,

$$\theta_{prior} = \mu. \tag{66}$$

$\mathbf{C}_D$ is a diagonal matrix given by [28]

$$\mathbf{C}_D = \sigma^2 \mathbf{I}, \tag{67}$$

**Table 4**
Matched data for parameter estimation study utilizing gradients computed via fine scale Adjoint Method.

| Well | Observed pressure [−] | Initial pressure [−] | Matched pressure [−] | Percent error [%] |
|---|---|---|---|---|
| OBSWELL1 | 0.2508 | 0.1693 | 0.2505 | 0.024 |
| OBSWELL2 | 0.6918 | 1.8305 | 0.6920 | 0.012 |

**Table 5**
Matched data for parameter estimation study utilizing gradients computed via fine scale Adjoint Method.

| Well | Observed pressure [−] | Initial pressure [−] | Matched pressure [−] | Percent error [%] |
|---|---|---|---|---|
| OBSWELL1 | 0.2508 | 0.1704 | 0.2498 | 0.080 |
| OBSWELL2 | 0.6918 | 0.8284 | 0.6929 | 0.065 |

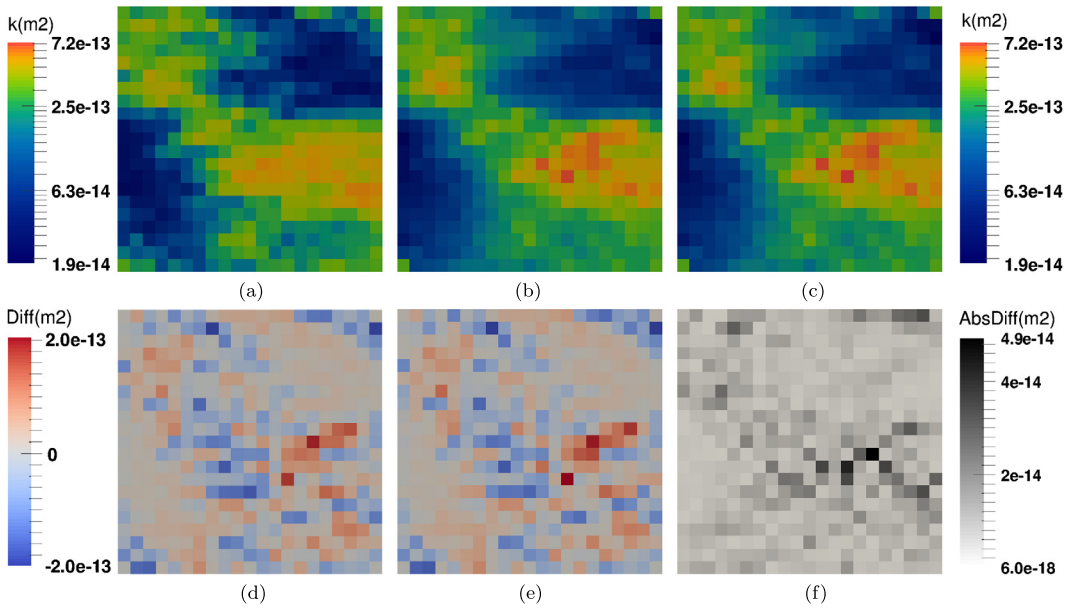


**Fig. 12.** Permeability field updates. Initial permeability field (a), after model calibration with fine scale (b) and MS (c) gradient computation. Difference between initial permeability field and fine scale (d) and MS (e) model calibration. Absolute difference between fine scale and MS permeability fields after model update is also shown (f).

where $\sigma^2$ is the variance of the data measurement error. In this experiment, the standard deviation of the pressure measurement error is $\sigma \approx 0.03$ (note that the measurement error is also non-dimensional). This represents a (very accurate) measurement error in the range of those usually employed in synthetic study cases (see e.g. [28]).

The optimization utilizes an LBGFS implementation [44], with a convergence criterion in the form of a small OF gradient norm value (10 $m^{-2}$). The matches obtained from the optimizations utilizing fine scale and MS Adjoint Methods are presented in Table 4 and Table 5, respectively.

It is clear that good matches are obtained via both gradient computation strategies. All matched pressure errors are in the order of $10^{-4}$, while the measurement errors are $10^{-5}$. The permeability fields estimated by both gradient methods are illustrated in Fig. 12. The same figure also shows the differences between initial and updated permeability fields, as well as the fine scale vs. MS estimated parameters.

It is clear that the updates employed when a MS gradient is provided (Fig. 12(e)) to the optimizer are close to the updates found when a fine scale gradient (Fig. 12(d)) is utilized.

Finally, the performance of the optimization algorithm is assessed when both (fine scale and MS) gradients are utilized. Fig. 13 illustrates the evolution of the normalized OF along the optimization process, where the OF values are normalized by its initial value.

Although MS-gradient approach is much more efficient (recall Fig. 2), and both optimizations lead to good matches when converged, the MS-gradient based optimization converges slower than the one based on the fine scale gradients. It is noted

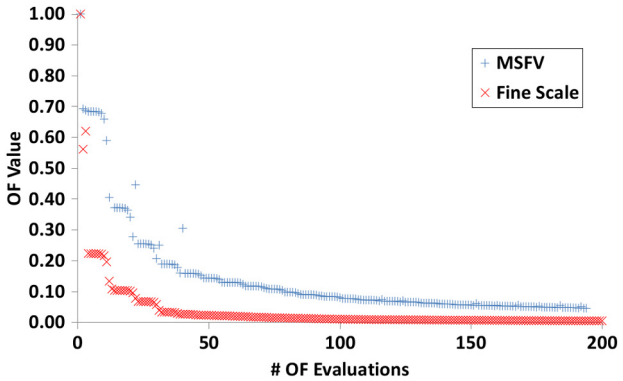


**Fig. 13.** Evolution of the normalized OF value for model calibration utilizing both fine scale and MSFV gradients.

that the quality of the MS solution can be further improved through an iterative procedure [14], which will be subject of our future research.

## 7. Conclusions

A Multiscale gradient computation strategy was developed based on a general algebraic formulation that does not depend on a particular objective function. This was possible by recasting the calculation of derivatives as multiplying a sensitivity matrix and its transpose with arbitrary matrices. The proposed framework is capable of providing different types of gradient information. Such flexibility allows the employment of the framework to any reservoir management study that requires gradient information. Also, the formulation naturally provides both Direct and Adjoint Methods. It was shown, via an asymptotic analysis, that the MS gradient computation strategy can be considerably more efficient than the fine scale strategy. Due to the algebraic nature of the formulation, the presented strategy can be applied to any MS (and multilevel) methodology.

The accuracy of the developed MS gradient computation strategy is studied for a set of examples with increasing complexity. The investigations show that the sources of inaccuracies in the MS solution (e.g. localization assumptions) also result in inaccuracies in the gradient computation. However, fortunately, strategies to improve the MS solution also improve the MS gradient accuracy. Another important observation is that greater angles between MS and fine scale gradient vectors are associated with small values of the gradient norms. Because small gradient norms indicate a weak association between parameters and responses, such differences are typically not very relevant in optimization. Lastly, in our example, the gradient directions, and therefore the convergence behavior of the gradient-based optimization procedure, were similar when either the fine scale or the MS strategy was applied. All this indicates that the presented method allows for accurate, yet less computationally expensive, gradient computations that can be successfully utilized in reservoir management studies.

Extension of the presented algorithm to include iterative multiscale strategies [14] is important for real-field applications. The key component of this extension is to develop an a-priori estimate of the quality of the gradients. Furthermore, for multiphase simulations, the introduced augmented state vector should also include the reconstruction of conservative field [52]. These are subjects of ongoing research.

## Acknowledgements

Rafael Moraes' PhD project was sponsored by Petrobras S.A. The authors would like to thank the Delft Advanced Reservoir Simulation (DARSim) research team for fruitful discussions during the development of the MS-gradient method.

## Appendix A. Tensors and tensor operations interpretation

The partial derivatives of **R**, **P** and **A** matrices with respect to the vector of parameters $\boldsymbol{\theta}$ result in third-order tensors. These tensors can be interpreted as a stack of matrices, with each matrix representing the partial derivatives of each of its columns with respect to the parameters. For example, a third order tensor defined by Eq. (39) reads

$$\frac{\partial \mathbf{P}}{\partial \boldsymbol{\theta}} = \left[ \frac{\partial \varphi_1}{\partial \boldsymbol{\theta}} \;\; \frac{\partial \varphi_2}{\partial \boldsymbol{\theta}} \;\; \cdots \;\; \frac{\partial \varphi_{N_C-1}}{\partial \boldsymbol{\theta}} \;\; \frac{\partial \varphi_{N_C}}{\partial \boldsymbol{\theta}} \right]_{N_F \times N_C \times N_\theta}, \tag{A.1}$$

where $\partial \mathbf{P}/\partial \boldsymbol{\theta}$ is an $N_F \times N_C \times N_\theta$ third-order tensor that can be interpreted as a stack of $N_\theta$ matrices of partial derivatives $\partial \boldsymbol{\varphi}_j/\partial \boldsymbol{\theta}$, $j = 1, \ldots, N_C$, each of dimension $N_F \times N_C$.

In order to illustrate the operations with this third-order tensor, the operations involved in the partial derivative Eq. (5) w.r.t. $\boldsymbol{\theta}$ are explained, disregarding the dependency of **R** on $\boldsymbol{\theta}$, i.e.,

$$\frac{\partial \check{\mathbf{g}}}{\partial \boldsymbol{\theta}}_{N_C\times N_\theta} = \mathbf{R}_{N_C\times N_F}\left(\overbrace{\frac{\partial \mathbf{A}}{\partial \boldsymbol{\theta}}_{N_F\times\boxed{N_F}\times N_\theta}\mathbf{P}_{\boxed{N_F}\times N_C}}^{1} + \underbrace{\mathbf{A}_{N_F\times\boxed{N_F}}\frac{\partial \mathbf{P}}{\partial \boldsymbol{\theta}}_{\boxed{N_F}\times N_C\times N_\theta}}_{2}\right)\check{\mathbf{x}}_{N_C}. \tag{A.2}$$

The subscripts represent the dimensions of the tensors. The product 1 in Eq. (A.2) can be interpreted as a stack of matrices resulting from $N_\theta$ products between matrices $\partial\mathbf{A}/\partial\boldsymbol{\theta}|_{:,:,k}$ of dimension $N_F \times N_F$ and the matrix $\mathbf{P}$ of dimension $N_F \times N_C$, $k = 1, ..., N_\theta$, and, therefore is a third-order tensor $N_F \times N_C \times N_\theta$. Analogously, product 2 also results in a third-order tensor of size $N_F \times N_C \times N_\theta$ as a stack of the products $\mathbf{A}$ times $\partial\mathbf{P}/\partial\boldsymbol{\theta}|_{:,:,k}$, $k = 1, ..., N_\theta$. In the equations, the "squared" dimensions highlight the dimensions which are operated on in each of the matrix products. Now consider the product of the term under parenthesis in Eq. (A.2) with the vector $\check{\mathbf{x}}$,

$$\frac{\partial \check{\mathbf{g}}}{\partial \boldsymbol{\theta}}_{N_C\times N_\theta} = \mathbf{R}_{N_C\times N_F}\underbrace{\left(\frac{\partial \mathbf{A}}{\partial \boldsymbol{\theta}}\mathbf{P} + \mathbf{A}\frac{\partial \mathbf{P}}{\partial \boldsymbol{\theta}}\right)_{N_F\times\boxed{N_C}\times N_\theta}\check{\mathbf{x}}_{\boxed{N_C}}}_{1}. \tag{A.3}$$

The product 1 in Eq. (A.3) can be interpreted as $N_\theta$ products between matrices $(\partial\mathbf{A}/\partial\boldsymbol{\theta}\mathbf{P} + \mathbf{A}\partial\mathbf{P}\partial\boldsymbol{\theta})|_{:,:,k}$ of dimensions $N_F \times N_C$ and the vector $\check{\mathbf{x}}$ of dimension $N_C$, $k = 1, ..., N_\theta$, hence leading to a matrix of dimensions $N_F \times N_\theta$, as follows

$$\frac{\partial \check{\mathbf{g}}}{\partial \boldsymbol{\theta}}_{N_C\times N_\theta} = \mathbf{R}_{N_C\times\boxed{N_F}}\left(\left(\frac{\partial \mathbf{A}}{\partial \boldsymbol{\theta}}\mathbf{P} + \mathbf{A}\frac{\partial \mathbf{P}}{\partial \boldsymbol{\theta}}\right)\check{\mathbf{x}}\right)_{\boxed{N_F}\times N_\theta}. \tag{A.4}$$

Finally, the last operation is a simple product between two matrices with the dimensions indicated in Eq. (A.4), leading to

$$\frac{\partial \check{\mathbf{g}}}{\partial \boldsymbol{\theta}}_{N_C\times N_\theta} = \left(\left(\mathbf{R}\left(\frac{\partial \mathbf{A}}{\partial \boldsymbol{\theta}}\mathbf{P} + \mathbf{A}\frac{\partial \mathbf{P}}{\partial \boldsymbol{\theta}}\right)\right)\check{\mathbf{x}}\right)_{N_C\times N_\theta}, \tag{A.5}$$

from where one can notice that the operations at the right-hand side lead to a matrix with the dimensions equal to those of the left-hand side matrix.

## References


[1] D.S. Oliver, Y. Chen, Recent progress on reservoir history matching: a review, Comput. Geosci. 15 (1) (2011) 185–221.
[2] J.D. Jansen, Adjoint-based optimization of multi-phase flow through porous media – a review, Comput. Fluids 46 (1) (2011) 40–51.
[3] B. Yeten, A. Castellini, B. Guyaguler, W. Chen, A comparison study on experimental design and response surface methodologies, in: SPE Reservoir Simulation Symposium, Society of Petroleum Engineers, 2005.
[4] L.J. Durlofsky, Upscaling and gridding of fine scale geological models for flow simulation, in: 8th International Forum on Reservoir Simulation, vol. 2024, Iles Borromees, Stresa, Italy, 2005.
[5] A. Datta-Gupta, M. King, Streamline Simulation: Theory and Practice, SPE Textbook Series, Society of Petroleum Engineers, 2007.
[6] J.D. Jansen, L.J. Durlofsky, Use of reduced-order models in well control optimization, Optim. Eng. (2016) 1–28, http://dx.doi.org/10.1007/s11081-016-9313-6.
[7] T.Y. Hou, X.-H. Wu, A multiscale finite element method for elliptic problems in composite materials and porous media, J. Comput. Phys. 134 (1) (1997) 169–189.
[8] P. Jenny, S.H. Lee, H.A. Tchelepi, Multi-scale finite-volume method for elliptic problems in subsurface flow simulation, J. Comput. Phys. 187 (1) (2003) 47–67.
[9] Y. Efendiev, T.Y. Hou, Multiscale Finite Element Methods: Theory and Applications, vol. 4, Springer Science & Business Media, 2009.
[10] S.H. Lee, H. Zhou, H. Tchelepi, Adaptive multiscale finite-volume method for nonlinear multiphase transport in heterogeneous formations, J. Comput. Phys. 228 (24) (2009) 9036–9058.
[11] H. Hajibeygi, P. Jenny, Adaptive iterative multiscale finite volume method, J. Comput. Phys. 230 (3) (2011) 628–643.
[12] A. Kozlova, Z. Li, J.R. Natvig, S. Watanabe, Y. Zhou, K. Bratvedt, S. Lee, A real-field multiscale black-oil reservoir simulator, in: SPE Reservoir Simulation Symposium, Society of Petroleum Engineers, 2015.
[13] M. Cusini, A.A. Lukyanov, J. Natvig, H. Hajibeygi, Constrained pressure residual multiscale (CPR-MS) method for fully implicit simulation of multiphase flow in porous media, J. Comput. Phys. 299 (2015) 472–486.
[14] H. Hajibeygi, G. Bonfigli, M.A. Hesse, P. Jenny, Iterative multiscale finite-volume method, J. Comput. Phys. 227 (19) (2008) 8604–8621.
[15] H. Zhou, H.A. Tchelepi, Two-stage algebraic multiscale linear solver for highly heterogeneous reservoir models, SPE J. 17 (02) (2012) 523–539.
[16] D. Cortinovis, P. Jenny, Iterative Galerkin-enriched multiscale finite-volume method, J. Comput. Phys. 277 (2014) 248–267.
[17] M. Ţene, M.S. Al Kobaisi, H. Hajibeygi, Algebraic multiscale method for flow in heterogeneous porous media with embedded discrete fractures (F-AMS), J. Comput. Phys. 321 (2016) 819–845.
[18] Y. Efendiev, S. Lee, G. Li, J. Yao, N. Zhang, Hierarchical multiscale modeling for flows in fractured media using generalized multiscale finite element method, GEM Int. J. Geomath. 6 (2) (2015) 141–162.
[19] H. Hajibeygi, H.A. Tchelepi, Compositional multiscale finite-volume formulation, SPE J. 19 (02) (2014) 316–326.
[20] P. Jenny, I. Lunati, Modeling complex wells with the multi-scale finite-volume method, J. Comput. Phys. 228 (3) (2009) 687–702.
[21] I. Lunati, P. Jenny, Multiscale finite-volume method for density-driven flow in porous media, Comput. Geosci. 12 (3) (2008) 337–350.
[22] Y. Wang, H. Hajibeygi, H.A. Tchelepi, Algebraic multiscale solver for flow in heterogeneous porous media, J. Comput. Phys. 259 (2014) 284–303.
[23] M. Ţene, Y. Wang, H. Hajibeygi, Adaptive algebraic multiscale solver for compressible flow in heterogeneous porous media, J. Comput. Phys. 300 (2015) 679–694.

[24] O. Møyner, K.-A. Lie, A multiscale restriction-smoothed basis method for high contrast porous media represented on unstructured grids, J. Comput. Phys. 304 (2016) 46–71.
[25] E. Parramore, M.G. Edwards, M. Pal, S. Lamine, Multiscale finite-volume CVD-MPFA formulations on structured and unstructured grids, Multiscale Model. Simul. 14 (2) (2016) 559–594.
[26] S. Shah, O. Møyner, M. Tene, K.-A. Lie, H. Hajibeygi, The multiscale restriction smoothed basis method for fractured porous media (F-MsRSB), J. Comput. Phys. 318 (2016) 36–57.
[27] M. Cusini, C. van Kruijsdijk, H. Hajibeygi, Algebraic dynamic multilevel (ADM) method for fully implicit simulations of multiphase flow in porous media, J. Comput. Phys. 314 (2016) 60–79.
[28] D.S. Oliver, A.C. Reynolds, N. Liu, Inverse Theory for Petroleum Reservoir Characterization and History Matching, Cambridge University Press, 2008.
[29] T.H. Michael, Scientific Computing: An Introductory Survey, The McGraw–Hill Companies Inc., New York, NY, USA, 2002.
[30] F. Anterion, R. Eymard, B. Karcher, Use of parameter gradients for reservoir history matching, in: SPE Symposium on Reservoir Simulation, Society of Petroleum Engineers, 1989.
[31] W.H. Chen, G.R. Gavalas, J.H. Seinfeld, M.L. Wasserman, A new algorithm for automatic history matching, SPE J. 14 (6) (1974) 593–608.
[32] G. Chavent, M. Dupuy, P. Lemmonier, History matching by use of optimal theory, SPE J. 15 (01) (1975) 74–86.
[33] R. Younis, K. Aziz, Parallel automatically differentiable data-types for next-generation simulator development, in: SPE Reservoir Simulation Symposium, Society of Petroleum Engineers, 2007.
[34] S.I. Aanonsen, G. Nævdal, D.S. Oliver, A.C. Reynolds, B. Vallès, The ensemble Kalman filter in reservoir engineering – a review, SPE J. 14 (03) (2009) 393–412.
[35] Y. Chen, D.S. Oliver, D. Zhang, Efficient ensemble-based closed-loop production optimization, SPE J. 14 (04) (2009) 634–645.
[36] R.M. Fonseca, B. Chen, J.D. Jansen, A.C. Reynolds, A stochastic simplex approximate gradient (StoSAG) for optimization under uncertainty, Int. J. Numer. Meth. Eng., http://dx.doi.org/10.1002/nme.5342.
[37] J. Fu, H.A. Tchelepi, J. Caers, A multiscale adjoint method to compute sensitivity coefficients for flow in heterogeneous porous media, Adv. Water Resour. 33 (6) (2010) 698–709.
[38] J. Fu, J. Caers, H.A. Tchelepi, A multiscale method for subsurface inverse modeling: single-phase transient flow, Adv. Water Resour. 34 (8) (2011) 967–979.
[39] S. Krogstad, V.L. Hauge, A. Gulbransen, Adjoint multiscale mixed finite elements, SPE J. 16 (01) (2011) 162–171.
[40] J.D. Jansen, A Systems Description of Flow Through Porous Media, Springer, 2013.
[41] H. Zhou, H.A. Tchelepi, Operator-based multiscale method for compressible flow, SPE J. 13 (02) (2008) 523–539.
[42] J.R.P. Rodrigues, Calculating derivatives for automatic history matching, Comput. Geosci. 10 (1) (2006) 119–136.
[43] J.F.B.M. Kraaijevanger, P.J.P. Egberts, J.R. Valstar, H.W. Buurman, Optimal waterflood design using the adjoint method, in: SPE Reservoir Simulation Symposium, Society of Petroleum Engineers, 2007.
[44] J. Nocedal, S. Wright, Numerical Optimization, Springer Science & Business Media, 2006.
[45] U. Trottenberg, C. Oosterlee, A. Schueller, Multigrid, Elsevier Academic Press, 2001.
[46] W.G.B. Smith, P. Bjorstad, Domain Decomposition, Cambridge University Press, 2004.
[47] J.E. Aarnes, Efficient domain decomposition methods for elliptic problems arising from flows in heterogeneous porous media, Comput. Vis. Sci. 8 (2) (2005) 93–106.
[48] N.M. Josuttis, C++ Templates: The Complete Guide, Addison–Wesley Professional, 2003.
[49] J.D. Jansen, A Simple Algorithm to Generate Small Geostatistical Ensembles for Subsurface Flow Simulation, Dept. of Geoscience and Engineering, Delft University of Technology, The Netherlands, 2013, Research Note, uuid:6000459e-a0cb-40d1-843b-81650053e093?collection=research.
[50] R.M. Fonseca, S.S. Kahrobaei, L.J.T. Van Gastel, O. Leeuwenburgh, J.D. Jansen, Quantification of the impact of ensemble size on the quality of an ensemble gradient using principles of hypothesis testing, in: SPE Reservoir Simulation Symposium, Society of Petroleum Engineers, 2015.
[51] N. Remy, A. Boucher, J. Wu, Applied Geostatistics with SGeMS: A User's Guide, Cambridge University Press, 2009.
[52] I. Lunati, P. Jenny, Multiscale finite-volume method for compressible multiphase flow in porous media, J. Comput. Phys. 216 (2) (2006) 616–636.