INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2011; **66**:976–999
Published online 1 March 2010 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/fld.2297

The independent set perturbation adjoint method: A new method of differentiating mesh-based fluids models

F. Fang[1,*,†], C. C. Pain[1], I. M. Navon[2], G. J. Gorman[1], M. D. Piggott[1] and P. A. Allison[1]

[1] *Applied Modelling and Computation Group, Department of Earth Science and Engineering, South Kensington Campus, Imperial College London, SW7 2AZ, U.K.*
[2] *Department of Scientific Computing, Florida State University, Tallahassee, FL 32306-4120, U.S.A.*

SUMMARY

A new scheme for differentiating complex mesh-based numerical models (e.g. finite element models), the Independent Set Perturbation Adjoint method (ISP-Adjoint), is presented. Differentiation of the matrices and source terms making up the discrete forward model is realized by a graph coloring approach (forming independent sets of variables) combined with a perturbation method to obtain gradients in numerical discretizations. This information is then convolved with the 'mathematical adjoint', which uses the transpose matrix of the discrete forward model. The adjoint code is simple to implement even with complex governing equations, discretization methods and non-linear parameterizations. Importantly, the adjoint code is independent of the implementation of the forward code. This greatly reduces the effort required to implement the adjoint model and maintain it as the forward model continues to be developed; as compared with more traditional approaches such as applying automatic differentiation tools. The approach can be readily extended to reduced-order models. The method is applied to a one-dimensional Burgers' equation problem, with a highly non-linear high-resolution discretization method, and to a two-dimensional, non-linear, reduced-order model of an idealized ocean gyre. Copyright © 2010 John Wiley & Sons, Ltd.

Received 28 August 2009; Revised 31 December 2009; Accepted 4 January 2010

KEY WORDS: automatic differentiation; optimization; adjoint; finite element; reduced-order models

1. INTRODUCTION

Adjoints to computational codes have been used for diverse applications including data assimilation [1–4], sensitivity analysis [5, 6], adaptive observations [7] and mesh-based error measures [8, 9]. However, differentiating discrete numerical models is generally impractical to perform by hand and remains challenging when using automatic differentiation (AD) tools ([10], more details on

[*] Correspondence to: F. Fang, Applied Modelling and Computation Group, Department of Earth Science and Engineering, South Kensington Campus, Imperial College London SW7 2AZ, U.K.

[†] E-mail: f.fang@imperial.ac.uk, URL: http://amcg.ese.imperial.ac.uk

Contract/grant sponsor: U.K.'s Natural Environment Research Council; contract/grant numbers: NER/A/S/2003/00595, NE/C52101X/1, NE/C51829X/1
Contract/grant sponsor: Engineering and Physical Sciences Research Council; contract/grant number: GR/R60898
Contract/grant sponsor: Leverhulme Trust; contract/grant number: F/07058/AB
Contract/grant sponsor: NURISP framework seven project
Contract/grant sponsor: Imperial College High Performance Computing Service
Contract/grant sponsor: Grantham Institute for Climate Change
Contract/grant sponsor: NSF; contract/grant numbers: ATM-0201808, CCF-0635162

AD are available at http://www.autodiff.org). Automatic differentiation also restricts the use of many advanced programming features, further increasing the difficulty of developing a forward and adjoint model. Both approaches often require intimate knowledge of the forward code in order to design a hand differentiation method or to alter the automatically produced code so that it works efficiently [11–14]. The time it takes to conduct this task may be greater than writing the forward model and there may also be computer science-related issues arising from the complexity of the computation of derivatives and memory usage [10, 15]. These challenges often preclude the use of gradient or adjoint variational (4D-VAR; three spatial dimensions plus time) data assimilation methods, sensitivity-based methods or any other method that uses gradients of key functionals with respect to the control variables (controls), e.g. material properties or initial/boundary conditions of flow models.

To reduce the computational cost and memory, Coleman and Moré [16] introduced the graph colouring scheme in the computation of derivatives. This allows for the efficient calculation of the gradient of a sparse matrix Jacobian (matrix containing key derivatives of the model w.r.t. the state variables) using AD software [17–19]. Other related research work and applications of graph coloring approaches in the numerical determination of large sparse derivative matrices (Jacobians) can be found in [20–24]. An overview of colouring schemes is provided in [22].

A forward model may have an actively developed dynamic core or utilize rapidly changing and complex parameterizations. Therefore, the application of consistent or exact gradient methods of differentiation to the discrete problem tends to lag behind developments in the forward model code. Despite this, there have been major successes in differentiating large models [25–29]. AD methods have been utilized on stable models at compile time after code modifications [30–33]. In addition, Marta *et al.* [10] developed a selective AD approach where AD is used to compute only certain terms of the discrete adjoint equations, and not to differentiate the entire solution algorithm.

The ISP-Adjoint approach, outlined here, is largely independent of implementation details of the forward model. The modelling software only needs to be modified so that it can accommodate the adjoint, i.e. the formation of the 'mathematical adjoint' transpose matrices of the forward problem, as well as the accommodation of the data structures necessary to provide access to the forward solution when time marching backwards in the gradient calculation. Similar approaches may be applied to reduced-order models such as proper orthogonal decomposition (POD) surrogates of full models.

The colouring approach, as shown here, can also be used to help accelerate the matrix equation assembly process on the assumption that the discretized system of equations has a polynomial representation and can thus be formed by a summation of pre-formed matrices. These matrices are formed before time marking or the non-linear iterations are begun and thus provides very rapid forward model assembly. They also enable differentiation to be easily performed, but again on the assumption that the discretized system of equations has a polynomial representation.

The remainder of this paper is set out as follows. The next section outlines the model governing equations followed by their discretization. Section 4 provides a description of how discrete system assembly may be performed using graph colouring methods. This is then followed by the extension to reduced-order POD models and the extension of the method to non-quadratic discrete systems of equations. In Section 5, the gradient of the discrete system (ISP-Adjoint) method is outlined and this is followed by an summary of the algorithm for forming the gradient based on the ISP-Adjoint. The method is applied to two problems: the Burgers' equation and the Navier–Stokes equations applied to ocean gyre circulation in Section 7 followed by discussion. Finally, conclusions are drawn.

2. GOVERNING EQUATIONS

Two systems of partial differential equations used to illustrate the application of the method are the Burgers' equation and the Navier–Stokes equations.

DOI: 10.1002/fld

The one-dimensional (1D) Burgers' equation is defined as

$$\frac{\partial u}{\partial t}+u\frac{\partial u}{\partial x}-v\frac{\partial^2 u}{\partial x^2}=0. \tag{1}$$

where u is the velocity, x is the spatial coordinate, t is the time coordinate and v is the viscosity coefficient.

The 3D Navier–Stokes continuity and momentum equations are considered in the form

$$\nabla\cdot\mathbf{v}=0, \tag{2}$$

$$\frac{\partial \mathbf{u}}{\partial t}=-\mathbf{u}\cdot\nabla\mathbf{u}-f\mathbf{k}\times\mathbf{u}+\nabla\cdot\boldsymbol{\tau}-\nabla p+\mathbf{s}_u, \tag{3}$$

where $\mathbf{u}=(u_x,u_y,u_z)^{\mathrm{T}}$ is the velocity vector, $\mathbf{x}=(x,y,z)^{\mathrm{T}}$ are the orthogonal Cartesian coordinates, $\hat{p}$ is the perturbation pressure ($p=p\prime/\rho_0$, ρ_0 is a constant reference density), $\mathbf{s}_u$ contains the body forces (wind stressed in the applications), f represents the Coriolis inertial force and $\mathbf{k}=(0,0,1)^{\mathrm{T}}$. The stress tensor $\boldsymbol{\tau}$ is used to represent viscous terms and are expressed in tensor form. The pressure variable is split into geostrophic and non-geostrophic parts that are solved separately. This allows for the accurate representation of geostrophic balance (for further details see [34]).

3. DISCRETE EQUATIONS

The discrete forward model representation of the governing equations above can be expressed in matrix form

$$\mathbf{A}\Phi=\mathbf{s}, \tag{4}$$

where $\Phi=(\Phi^{1^{\mathrm{T}}}\ \Phi^{2^{\mathrm{T}}}\ \dots\ \Phi^{\mathcal{N}_t{}^{\mathrm{T}}})^{\mathrm{T}}$ is the vector of state variables, $\mathbf{s}=(\mathbf{s}^{1^{\mathrm{T}}}\ \mathbf{s}^{2^{\mathrm{T}}}\ \dots\ \mathbf{s}^{\mathcal{N}_t{}^{\mathrm{T}}})^{\mathrm{T}}$, $\mathbf{A}$ is the global matrix making up the discretization in the forward model at all the time levels where $\mathcal{N}_t$ is the number of time levels. At each time level n, the state and source vectors are $\Phi^n=(\Phi^n_1\ \Phi^n_2\ \dots\ \Phi^n_{\mathcal{N}})^{\mathrm{T}}$ and $\mathbf{s}^n=(s^n_1\ s^n_2\ \dots\ s^n_{\mathcal{N}})^{\mathrm{T}}$, respectively, in which Φ^n_i and s^n_i are scalars. In addition, $\mathcal{N}$ is the number of unknowns solved for at each time level and for steady-state problems $\mathcal{N}_t=1$.

For the two-level time marching methods used here, the global matrix $\mathbf{A}$ has the structure

$$\mathbf{A}=\begin{pmatrix}\mathbf{P}^1 & & & \\ \mathbf{H}^2 & \mathbf{P}^2 & & \\ & \ddots & \ddots & \\ & & \mathbf{H}^{\mathcal{N}_t} & \mathbf{P}^{\mathcal{N}_t}\end{pmatrix}. \tag{5}$$

3.1. Burgers' equation

A backward Euler time stepping method at time level n and centered on control volume (CV) i for the Burgers' equation (1) can result in the following schemes (u_i is an approximation to u in CV i):

1. With central differencing in space

$$\frac{u_i^{n+1}-u_i^n}{\Delta t}+u_i^n\frac{u_{i+1}^{n+1}-u_{i-1}^{n+1}}{2\Delta x}+v\frac{-u_{i+1}^{n+1}+2u_i^{n+1}-u_{i-1}^{n+1}}{(\Delta x)^2}=0; \tag{6}$$

2. With first-order upwind discretization in space:

$$\begin{aligned}&\frac{u_i^{n+1}-u_i^n}{\Delta t}-\max\left(0,\frac{1}{2\Delta x}(u_i^n+u_{i-1}^n)\right)u_{i-1}^{n+1}-\min\left(0,\frac{1}{2\Delta x}(u_i^n+u_{i-1}^n)\right)u_i^{n+1}\\&\quad+\max\left(0,\frac{1}{2\Delta x}(u_{i+1}^n+u_i^n)\right)u_i^{n+1}+\min\left(0,\frac{1}{2\Delta x}(u_{i+1}^n+u_i^n)\right)u_{i+1}^{n+1}\\&\quad-\frac{u_{i+1}^n-u_{i-1}^n}{2\Delta x}u_i^{n+1}+v\frac{-u_{i+1}^{n+1}+2u_i^{n+1}-u_{i-1}^{n+1}}{(\Delta x)^2}=0,\end{aligned}\tag{7}$$

3. With high-resolution discretization in space:

$$\begin{aligned}&\frac{u_i^{n+1}-u_i^n}{\Delta t}-u_{i-1/2}^n(\xi_{i-1/2}^n u_{i-1}^{n+1}+(1-\xi_{i-1/2}^n)u_i^{n+1})+u_{i+1/2}^n(\xi_{i+1/2}^n u_i^{n+1}+(1-\xi_{i+1/2}^n)u_{i+1}^{n+1})\\&\quad-\frac{u_{i+1}^n-u_{i-1}^n}{2\Delta x}u_i^{n+1}+v\frac{-u_{i+1}^{n+1}+2u_i^{n+1}-u_{i-1}^{n+1}}{(\Delta x)^2}=0,\end{aligned}\tag{8}$$

in which Δx is the size of the CV's.

To determine in the high-resolution method where to apply first-order instead of high-order fluxes, there is a need to be able to detect an extrema and if there is to be a smooth transition between the two fluxes then there is a need to quantify how close to an extrema the solution is. This is achieved here using the normalized variable diagram NVD method [35]. Suppose u_u^n, u_c^n, u_d^n are ordered consecutively and u_f^n is the face value between CV's c and d. If $f=i-1/2$ and $u_f^n>0$ then subscripts $u=i-2$, $c=i-1$, $d=i$ and if $u_f^n<0$ then subscripts $u=i+1$, $c=i$, $d=i-1$ and $u_{i-1/2}^n=\frac{1}{2}(u_{i-1}^n+u_i^n)$. For a monotonic solution,

$$\mathscr{u}_f^n=\frac{u_f^n-u_u^n}{u_d^n-u_u^n}\in[0,1],\tag{9}$$

which is the non-dimensional high-order face value. The non-dimensional upwind face value is

$$\mathscr{u}_c^n=\frac{u_c^n-u_u^n}{u_d^n-u_u^n}.\tag{10}$$

The idea is to obtain a linear combination $\widetilde{\mathscr{u}}_f^n$ of $\mathscr{u}_f^n$ and $\mathscr{u}_c^n$ such that $\widetilde{\mathscr{u}}_f^n$ equals $\mathscr{u}_c^n$ when there exists an extrema ($\mathscr{u}_c^n\notin[0,1]$ and the scheme becomes first order) and $\widetilde{\mathscr{u}}_f^n$ moves smoothly between this and the high-order flux $\mathscr{u}_f^n$. $\widetilde{u}_f^n$ is calculated from

$$\widetilde{\mathscr{u}}_f^n=\frac{\widetilde{u}_f^n-u_u^n}{u_d^n-u_u^n}.\tag{11}$$

The flux limited solution is

$$\widetilde{u}_f^n=\widetilde{\mathscr{u}}_f^n(u_d^n-u_u^n)+u_u^n.\tag{12}$$

$\widetilde{\mathscr{u}}_f^n$ in this equation is calculated from the normalized variable diagram, that is

$$\widetilde{\mathscr{u}}_f^n=\begin{cases}\min\{1,2\mathscr{u}_c^n,\max\{\mathscr{u}_c^n,\mathscr{u}_f^n\}\} & \text{if } \mathscr{u}_c^n\in(0,1);\\ \mathscr{u}_c^n & \text{otherwise.}\end{cases}\tag{13}$$

Thus, at an extrema the first-order non-oscillatory method will be applied:

$$\xi_f^n = \begin{cases} \dfrac{1}{2}\left(1 - \dfrac{\widetilde{u}_f^n - u_c^n}{u_f^n - u_c^n}\right) + \dfrac{1}{2} & \text{if } u_f^n > 0; \\[2ex] 1 - \left(\dfrac{1}{2}\left(1 - \dfrac{\widetilde{u}_f^n - u_c^n}{u_f^n - u_c^n}\right) + \dfrac{1}{2}\right) & \text{otherwise.} \end{cases}$$

3.2. Navier–Stokes equations

The variables (u_x, u_y, u_z, p) at time level n are approximated with a finite element representation:

$$u_x^n = \sum_{j=1}^{\mathcal{N}_u} N_j {u_x}_j^n, \quad u_y^n = \sum_{j=1}^{\mathcal{N}_u} N_j {u_y}_j^n, \quad u_z^n = \sum_{j=1}^{\mathcal{N}_u} N_j {u_z}_j^n, \quad p^n = \sum_{j=1}^{\mathcal{M}} M_j p_j^n, \tag{14}$$

where N_j and M_j are finite element basis functions associated with velocity and pressure, respectively, and with corresponding number of basis functions $\mathcal{N}_u$ and $\mathcal{M}$. The discrete model for incompressible fluids' (Equations (2) and (3)) calculations can be written for finite element methods as explained in [34] as

$$\mathbf{C}^{n+1\mathrm{T}}\mathbf{u}^{n+1} = 0, \tag{15}$$

$$\mathbf{E}^{n+1}\mathbf{u}^{n+1} = \mathbf{B}^{n+1}\mathbf{u}^n + \mathbf{C}^{n+1}\mathbf{p}^{n+1} + \mathbf{s}^{n+1}, \tag{16}$$

where n is the time level and $\mathbf{s}$ includes the discretized sources, initial and boundary conditions and body forces. The vector of velocities is defined as

$$\mathbf{u}^n = (\mathbf{u}_x^{n\mathrm{T}}, \mathbf{u}_y^{n\mathrm{T}}, \mathbf{u}_z^{n\mathrm{T}})^{\mathrm{T}},$$

with sub-vectors associated with the velocity components,

$$\mathbf{u}_x^n = ({u_x}_1^n, {u_x}_2^n, \ldots, {u_x}_{\mathcal{N}_u}^n)^{\mathrm{T}},$$

$$\mathbf{u}_y^n = ({u_y}_1^n, {u_y}_2^n, \ldots, {u_y}_{\mathcal{N}_u}^n)^{\mathrm{T}},$$

$$\mathbf{u}_z^n = ({u_z}_1^n, {u_z}_2^n, \ldots, {u_z}_{\mathcal{N}_u}^n)^{\mathrm{T}},$$

where all subscripts indicate nodal values of the associated variables. The pressure vector is

$$\mathbf{p}^{n+1} = (p_1^{n+1}, p_2^{n+1}, \ldots, p_{\mathcal{M}}^{n+1})^{\mathrm{T}}.$$

The matrices $\mathbf{E}^{n+1}$ and $\mathbf{B}^{n+1}$ are non-linear in $\mathbf{u}$ and $\mathbf{C}^{n+1}$ is the matrix associated with the pressure vector.

For the Navier–Stokes discrete equations (15) and (16), the vector of state variables at time level n is $\Phi^n = (\mathbf{u}^{n\mathrm{T}}, \mathbf{p}^{n\mathrm{T}})^{\mathrm{T}}$. Given the structure of matrix $\mathbf{A}$ Equation (5) the matrix that is solved at time level n typically has a structure that is a block matrix representation of the discrete system (15):

$$\mathbf{P}^n = \begin{pmatrix} \mathbf{E}^n & \mathbf{C}^n \\ \mathbf{C}^{n\mathrm{T}} & 0 \end{pmatrix} \tag{17}$$

and

$$\mathbf{H}^n = \begin{pmatrix} \mathbf{B}^n & 0 \\ 0 & 0 \end{pmatrix}. \tag{18}$$

4. DISCRETE SYSTEM ASSEMBLY USING GRAPH COLOURING METHODS

This section provides a description of how discrete system assembly can be performed using graph colouring methods for a discrete polynomial system (polynomial in the state variables). The matrices associated with the non-linear terms can be expressed by a set of sub-matrices. For time-dependent problems these sub-matrices are time-independent and can be easily differentiated to form adjoint systems of equations (this will be discussed in the following section). This approach is also extended to POD reduced-order modelling and in this case the number of POD bases is equal to the number of colors used in the graph colouring method.

4.1. Graph colouring

Graph colouring methods are commonly used to model the dependency between different subtasks or data. Here we define the graph $G_r=(V_{\mathrm{g}}, E_{\mathrm{g}})$, where the vertex set, V_{g}, are the nodes or cells of the finite element or CV mesh (i.e. the rows of a discretization matrix), and the edge set, E_{g}, is defined by the connectivity under a given stencil, see below. The chromatic number, $\chi(G_r)$, is bounded by

$$\omega(G_r)\leqslant\chi(G_r)\leqslant\Delta(G_r)+1,$$

where $\omega(G_r)$ is the clique number and $\Delta(G_r)$ is the maximum vertex degree [36]. $\chi(G_r)$ is the minimum number of colors necessary to color a graph. However, the number of colors obtained by a colouring algorithm $\mathcal{N}_c$ might be greater than this minimum.

When forming non-linear discretizations like $\int_\Omega N_i q N_j \,\mathrm{d}\Omega$, for some $q=\sum_{k=1}^{\mathcal{Q}} Q_k q_k$ and domain Ω, it is important to look at the independent sets of basis functions Q_k used in $\int_\Omega N_i Q_k N_j \,\mathrm{d}\Omega$ for any nodes i and j. Q_k ($1\leqslant k\leqslant\mathcal{Q}$, $\mathcal{Q}$ is the number of basis functions Q_k) is the finite basis function of a variable q. That is for the ith row and jth column of the matrix associated with $\int_\Omega N_i Q_k N_j \,\mathrm{d}\Omega$ no two basis functions Q_k of the same color contribute a non-zero value to this row and column. The colouring scheme (dependence) is shown in Figure 1. In practice for a node-wise Q_k shown in Figure 1(b), the graph associated with the non-zeros of the matrix $\mathbf{M}^{\mathrm{T}}\mathbf{M}$ (for example, the linear matrix that has the element of $M_{i,j}=\int_\Omega N_i N_j \,\mathrm{d}\Omega$ for nodes i and j) can be colored to achieve the required independent sets of $\int_\Omega N_i Q_k N_j \,\mathrm{d}\Omega$. This is also known as a distance-2 colouring of the sparsity pattern of matrix $\mathbf{M}$ [22]. A typical colouring is also shown in Figure 1(b). Thus depending on the form of Q_k, the matrix stencil and the vertex degree of the graph will vary. This would be the case for a material property or non-linearity that would be expanded as $q=\sum_{k=1}^{\mathcal{Q}} q_k Q_k$ where $\mathbf{q}=(q_1, q_2, \ldots, q_{\mathcal{Q}})^{\mathrm{T}}$. For the case where Q_k is constant throughout an element (element-wise) and for the case when Q_k has a bi-linear node-wise variation, the colouring needs to be chosen as shown in Figures 1(a) and (b), respectively.

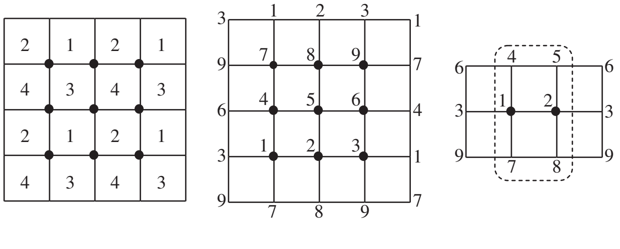


Figure 1. Extended colouring for material properties and non-linearities. Left: graph colouring scheme for an element-wise material property or non-linearity. Four colors are needed for 2D linear quadrilateral elements. Middle: graph colouring scheme for a node-wise material property or non-linearity. Nine colors are needed for 2D linear quadrilateral elements with a distance-2 colouring. Right: for the case shown in (b), the dotted line indicates the nodes that are directly linked to both nodes 1 and 2. These are the nodes for which there needs to be an independent set and they must be colored with a different number than 1 and 2.

4.2. Matrix representation of non-linear discrete systems

For a non-linearity with a polynomial representation in the discrete system (e.g. the Navier–Stokes equations discretized with a continuous Bubnov Galerkin method), the matrix equations can be formed from a summation of matrices. For illustration, the following are considered here: quadratic discrete systems, high-order polynomial discrete systems and non-polynomial systems.

4.2.1. Quadratic discrete systems. As discussed above, an example of a quadratic non-linearity in the can be written in a matrix of the form

$$\mathbf{Q}_{q\,ij} = \int_{\Omega} N_i q N_j \,\mathrm{d}\Omega, \tag{19}$$

where q is a variable, for example, the velocity. The application to other finite element matrices, such as those formed from inertia or advection terms of the form $\int_{\Omega} N_i u_x^n \partial N_j/\partial x \,\mathrm{d}\Omega$, follows along similar lines, in which $q = u_x^n$. The matrix $\mathbf{Q}_{q\,ij}^{\;c}$ in the colouring scheme can be expressed as

$$\mathbf{Q}_{q\,ij}^{\;c} = \int_{\Omega} N_i b^c N_j \,\mathrm{d}\Omega, \tag{20}$$

where

$$b^c = \sum_{k=1}^{\mathscr{Q}} Q_k b_k^c, \quad b_k^c = \begin{cases} 1 & \text{if node } k \text{ is of colour } c, \\ 0 & \text{at other nodes.} \end{cases} \tag{21}$$

Applying the same approach as outlined here for matrix $\mathbf{Q}$ to matrices $\mathbf{E}^{n+1}$ and $\mathbf{B}^{n+1}$ in Equation (16) means that the methods outlined here to efficiently construct the matrix $\mathbf{Q}$ may also be applied to form the matrices $\mathbf{E}^{n+1}$ and $\mathbf{B}^{n+1}$. The matrix $\mathbf{Q}$ can be constructed by a set of sub-matrices using graph colouring,

$$\mathbf{Q}_{q\,ij} = \sum_{c=1}^{\mathscr{N}_c} \mathbf{Q}_{q\,ij}^{c} q_{ij}^{c} = \overline{\mathbf{Q}}_{q\,ij} + \sum_{c=1}^{\mathscr{N}_c} \mathbf{Q}_{q\,ij}^{c} (q_{ij}^{c} - \bar{q}_{ij}^{c}), \tag{22}$$

where q_{ij}^c is the value of q at a node of color c neighboring both nodes i and j (see Figure 1(c)), $\mathscr{N}_c$ is the number of colors. $\mathscr{N}_c$ is four for 2D linear element cases, see Figure 1(b). This graph is colored with nine colors, which are needed to form the independent sets in this work and form a distance-2 colouring. Note that it is useful to write the matrix (22) as a perturbation from a mean matrix $\overline{\mathbf{Q}}_{q\,ij} = \int_{\Omega} N_i \bar{q} N_j \,\mathrm{d}\Omega = \sum_{c=1}^{\mathscr{N}_c} \mathbf{Q}_{q\,ij}^{c} \bar{q}_{ij}^{c}$ in order to linearize about a solution $\bar{q} = \sum_{k=1}^{\mathscr{Q}} Q_k \bar{q}_k$, e.g. the mean of q or the most recently available value of q.

For a quadratic discrete model like a finite element Bubnov–Galerkin discretization of the Navier–Stokes equations [37], the matrices $\overline{\mathbf{Q}}_q$, $\mathbf{Q}_q^c$ are time-independent. In practice, the matrices $\mathbf{E}^{n+1}$ and $\mathbf{B}^{n+1}$ in Equation (16) would need to be formed in this way. The matrices $\overline{\mathbf{Q}}_q$, $\mathbf{Q}_q^c$ can be constructed by sending down different values of the non-linear terms into the matrix assembly routines.

4.2.2. High-order polynomial discrete systems. For a high-order discrete model, e.g. cubic discrete model, the matrix $\mathbf{Q}_{uq}$ can be written as

$$\mathbf{Q}_{rq\,ij} = \int_{\Omega} N_i r q N_j \,\mathrm{d}\Omega, \tag{23}$$

where the velocity $q = \sum_{k=1}^{\mathscr{Q}} Q_k q_k$. The above is repeated but assuming a specified r. That is the matrix $\mathbf{Q}_r$ is formed using the colouring method of the previous section applied to $\mathbf{Q}_q$. Then the same method is applied to this matrix as outlined previously to form the $\mathbf{Q}_{rq}$. Thus

$$\mathbf{Q}_{rq\,ij} = \overline{\mathbf{Q}}_{rq\,ij} + \sum_{c=1}^{\mathscr{N}_c} \mathbf{Q}_{rq\,ij}^{c} (q_{ij}^{c} - \bar{q}_{ij}^{c}), \tag{24}$$

where

$$\mathbf{Q}_{rq}{}^{c}_{ij}=\int_{\Omega} N_i r b^c N_j \,\mathrm{d}\Omega, \quad \overline{\mathbf{Q}}_{rq}{}^{c}_{ij}=\int_{\Omega} N_i r \bar{q} N_j \,\mathrm{d}\Omega. \tag{25}$$

4.2.3. Non-polynomial systems. The extension to non-polynomial discrete systems, which may be obtained for certain discretization methods (e.g. upwind methods, discontinuous Galerkin, flux limiting methods, or model parameterizations such as large eddy simulation (LES) methods), is realized by linearizing the system about $\bar{q}$ say, if q is the non-linear variable. The linearization realized through $\bar{q}$ may have to be re-performed often throughout a time-dependent simulation in order to achieve accuracy. Thus, the matrices may be formed by a summation of matrices. This makes the matrices easy to differentiate and fast to assemble, but it involves an approximation. A similar approach may also be applied to reduced-order models to help in the rapid assembly of their equations as described below.

4.3. Discrete system of reduced-order model equations

The aim behind reduced-order models is to find a relatively small number of basis functions (few relative to the full model) that can accurately represent the system dynamics. These basis functions are often obtained from the full forward model results by applying a singular value decomposition analysis to a number of snapshots of the forward solution to determine the most energetic modes of the system and in this way optimally obtain basis functions. This is the approach taken in POD to obtain the basis functions, see [38]. In this section, the matrices are generated by a summation of matrices similar to the colouring method used above for the finite element equations.

For the Navier–Stokes model for example, once the reduced-order basis functions are obtained the velocity variables $\mathbf{u}$ can be expressed as an expansion of the POD basis functions $\{\Psi_{u_x 1},\ldots,\Psi_{u_x \mathscr{P}_u}\}$, $\{\Psi_{u_y 1},\ldots,\Psi_{u_y \mathscr{P}_u}\}$ and $\{\Psi_{u_z 1},\ldots,\Psi_{u_z \mathscr{P}_u}\}$ (here $\mathscr{P}_u$ is the number of POD bases for velocity):

$$\begin{pmatrix} u_x(t,x,y,z)\\ u_y(t,x,y,z)\\ u_z(t,x,y,z)\end{pmatrix}=\begin{pmatrix}\bar{u}_x\\ \bar{u}_y\\ \bar{u}_z\end{pmatrix}+\sum_{k=1}^{\mathscr{P}_u}\mathbf{N}_k^{\boldsymbol{\Psi}_\mathbf{u}}(\mathbf{x})\mathbf{a}_{\mathbf{u}\,k}(t), \tag{26}$$

and pressure is decomposed into $p=p_{\mathrm{ng}}+p_{\mathrm{g}}$ in which p_{ng}, p_{g} are the non-geostrophic and geostrophic pressures (see [38] for details associated with p_{g}) and the POD basis functions $\{\Psi_{p1},\ldots,\Psi_{p\mathscr{P}_p}\}$ (here $\mathscr{P}_p$ is the number of POD bases) for non-geostrophic pressure are used to obtain

$$p_{\mathrm{ng}}(t,x,y,z)=\bar{p}_{\mathrm{ng}}+\sum_{k=1}^{\mathscr{P}_p}\Psi_{p\,k}(\mathbf{x})a_{p\,k}(t), \tag{27}$$

whereas the geostrophic pressure can be represented by a summation of the two sets of geostrophic basis functions $\Psi_{\mathrm{g}u_x k}$ and $\Psi_{\mathrm{g}u_y k}$, which are obtained by $\Psi_{u_x k}$ and $\Psi_{u_y k}$ [38]:

$$p_{\mathrm{g}}(t,x,y,z)=\bar{p}_{\mathrm{g}}+\sum_{k=1}^{\mathscr{P}_u}\Psi_{\mathrm{g}u_x k}(\mathbf{x})a_{u_x k}(t)+\sum_{k=1}^{\mathscr{P}_u}\Psi_{\mathrm{g}u_y k}(\mathbf{x})a_{u_y k}(t), \tag{28}$$

where $\mathbf{a}_{\mathbf{u}\,k}=(a_{u_x k},a_{u_y k},a_{u_z k})^{\mathrm{T}}$ and $a_{p\,k}$ are the coefficients to be determined for velocity $(u_x,u_y,u_z)^{\mathrm{T}}$ and p, respectively, $\bar{p}_{\mathrm{ng}}$, $\bar{p}_{\mathrm{g}}$ and $(\bar{u}_x,\bar{u}_y,\bar{u}_z)^{\mathrm{T}}$ are the means of the ensemble of snapshots for the variables p_{ng}, p_{g} and $(u_x,u_y,u_z)^{\mathrm{T}}$, respectively, $\mathbf{N}_k^{\boldsymbol{\Psi}_\mathbf{u}}$ is a diagonal matrix including the POD basis for the velocity variable $(u_x,u_y,u_z)^{\mathrm{T}}$, in the finite element method, the POD basis $\boldsymbol{\Psi}_{\mathbf{u}\,k}(\mathbf{x})=\sum_{i=1}^{\mathscr{N}_u}N_i\boldsymbol{\Psi}_{\mathbf{u}\,k\,i}$ and $\Psi_{p\,k}(\mathbf{x})=\sum_{i=1}^{\mathscr{N}_u}N_i\Psi_{p\,k\,i}$, where N_i is the basis function in the finite element and $\mathscr{N}_u$ is the number of velocity nodes in the computational domain.

DOI: 10.1002/fld

Substituting Equations (26), (27) and (28) into (2) and (3) and taking the POD basis functions $\boldsymbol{\Psi}_p$ and $\boldsymbol{\Psi}_u$ as the test function for (2) and (3), respectively, then integrating over the computational domain Ω,

$$\int_\Omega \mathbf{N}_l^p F_p\left(\begin{pmatrix}\bar{u}_x\\ \bar{u}_y\\ \bar{u}_z\end{pmatrix}+\sum_{k=1}^{\mathscr{P}_u}\mathbf{N}_k^{\boldsymbol{\Psi}_\mathbf{u}}(\mathbf{x})\mathbf{a}_{\mathbf{u}\,k},\, p,\, t,\, \mathbf{x}\right)\mathrm{d}\Omega=0, \tag{29}$$

$$\begin{aligned}\int_\Omega \mathbf{N}_l^{\boldsymbol{\Psi}_\mathbf{u}}\frac{\partial}{\partial t}\left(\begin{pmatrix}\bar{u}_x\\ \bar{u}_y\\ \bar{u}_z\end{pmatrix}+\sum_{k=1}^{\mathscr{P}_u}\mathbf{N}_k^{\boldsymbol{\Psi}_\mathbf{u}}(\mathbf{x})\mathbf{a}_{\mathbf{u}\,k},\right)\mathrm{d}\Omega\\ =\int_\Omega \mathbf{N}_l^{\boldsymbol{\Psi}_\mathbf{u}}F_\mathbf{u}\left(\begin{pmatrix}\bar{u}_x\\ \bar{u}_y\\ \bar{u}_z\end{pmatrix}+\sum_{k=1}^{\mathscr{P}_u}\mathbf{N}_k^{\boldsymbol{\Psi}_\mathbf{u}}(\mathbf{x})\mathbf{a}_{\mathbf{u}\,k},\, p,\, t,\, \mathbf{x}\right)\mathrm{d}\Omega,\end{aligned} \tag{30}$$

where F_p is defined as the left-hand side of (2), whereas $F_\mathbf{u}$ is defined as the right-hand side of (3), $\mathbf{N}_l^p=\Psi_{p\,l}$ for the continuity Equation (2) and $\mathbf{N}_l^{\boldsymbol{\Psi}_\mathbf{u}}=\mathrm{diag}(\Psi_{u_x\,l},\Psi_{u_y\,l},\Psi_{u_z\,l})$ for the Navier–Stokes equation (3).

The POD reduced-order model is then obtained:

$$\left\langle N_l^{\Psi_p},\nabla\cdot\left(\begin{pmatrix}\bar{u}_x\\ \bar{u}_y\\ \bar{u}_z\end{pmatrix}+\sum_{k=1}^{\mathscr{P}_u}\mathbf{N}_k^{\boldsymbol{\Psi}_\mathbf{u}}(\mathbf{x})\mathbf{a}_{\mathbf{u}\,k}(t)\right)\right\rangle=0, \tag{31}$$

and

$$\begin{aligned}&\frac{\partial\mathbf{a}_{\mathbf{u}l}(t)}{\partial t}+\left\langle\mathbf{N}_l^{\boldsymbol{\Psi}_\mathbf{u}},\left(\left(\begin{pmatrix}\bar{u}_x\\ \bar{u}_y\\ \bar{u}_z\end{pmatrix}+\sum_{k=1}^{\mathscr{P}_u}\mathbf{N}_k^{\boldsymbol{\Psi}_\mathbf{u}}(\mathbf{x})\mathbf{a}_{\mathbf{u}\,k}(t)\right)\cdot\nabla\left(\begin{pmatrix}\bar{u}_x\\ \bar{u}_y\\ \bar{u}_z\end{pmatrix}+\sum_{k=1}^{\mathscr{P}_u}\mathbf{N}_k^{\boldsymbol{\Psi}_\mathbf{u}}\mathbf{a}_{\mathbf{u}\,k}(t)\right)\right)\right\rangle\\ &+\left\langle\mathbf{N}_l^{\boldsymbol{\Psi}_\mathbf{u}},\left(f\mathbf{k}\times\left(\begin{pmatrix}\bar{u}_x\\ \bar{u}_y\\ \bar{u}_z\end{pmatrix}+\sum_{k=1}^{\mathscr{P}_u}\mathbf{N}_k^{\boldsymbol{\Psi}_\mathbf{u}}(\mathbf{x})\mathbf{a}_{\mathbf{u}\,k}(t)\right)\right)\right\rangle\\ &+\left\langle\mathbf{N}_l^{\boldsymbol{\Psi}_\mathbf{u}},\left(\nabla p-\mu\nabla^2\left(\begin{pmatrix}\bar{u}_x\\ \bar{u}_y\\ \bar{u}_z\end{pmatrix}+\sum_{k=1}^{\mathscr{P}_u}\mathbf{N}_k^{\boldsymbol{\Psi}_\mathbf{u}}(\mathbf{x})\mathbf{a}_{\mathbf{u}\,k}(t)\right)\right)\right\rangle=0,\end{aligned} \tag{32}$$

subject to the initial condition

$$\mathbf{a}_{\mathbf{u}\,l}(0)=\left\langle\begin{pmatrix}u_x^0-\bar{u}_x\\ u_y^0-\bar{u}_y\\ u_z^0-\bar{u}_z\end{pmatrix},\mathbf{N}_l^{\boldsymbol{\Psi}_\mathbf{u}}\right\rangle, \tag{33}$$

where $<\cdot,\cdot>$ is the canonical inner product in the L^2 norm, $\mathbf{a}=(\mathbf{a}_\mathbf{u}^\mathrm{T},a_p)^\mathrm{T}$, $1\leqslant l\leqslant\mathscr{P}_p$ for the continuity equation (2) whereas $1\leqslant l\leqslant\mathscr{P}_u$ for the momentum equation (3).

The discrete model of (31) and (32) at the time level $n+1$ can be written in a general form in a subspace:

$$\mathbf{C}_{\mathrm{POD}}^{n+1\mathrm{T}}\mathbf{a}_{\mathbf{u}}^{n+1}=0, \tag{34}$$

and

$$\mathbf{E}_{\mathrm{POD}}^{n+1}\mathbf{a}_{\mathbf{u}}^{n+1}=\mathbf{B}_{\mathrm{POD}}^{n+1}\mathbf{a}_{\mathbf{u}}^{n}+\mathbf{C}_{\mathrm{POD}}^{n+1}\mathbf{a}_{p}^{n+1}+\mathbf{s}_{\mathrm{POD}}^{n+1}, \tag{35}$$

where $\mathbf{E}_{\mathrm{POD}}^{n+1}$, $\mathbf{B}_{\mathrm{POD}}^{n+1}$ (associated with the non-linear term) and $\mathbf{C}_{\mathrm{POD}}^{n+1}$ (associated with the pressure term) are the matrices including all the discretization of (31) and (32), $\mathbf{s}_{\mathrm{POD}}^{n+1}$ is a discretized source term.

The (k,l)th (here $1\leqslant k,\ l\leqslant 3\mathscr{P}_u$) entry of the matrices $\mathbf{E}_{\mathrm{POD}}^{n+1}$ and $\mathbf{B}_{\mathrm{POD}}^{n+1}$ can be constructed by the entries of the matrices $\mathbf{E}^{n+1}$ and $\mathbf{B}^{n+1}$ in the full model (16):

$$\mathbf{E}_{\mathrm{POD}\,kl}^{n+1}=\sum_{i=1}^{3\mathscr{N}_{\mathbf{u}}}\sum_{j=1}^{3\mathscr{N}_{\mathbf{u}}}\mathbf{E}_{ij}^{n+1}\mathbf{D}_{ki}^{\mathbf{\Psi}_{\mathbf{u}}}\mathbf{D}_{lj}^{\mathbf{\Psi}_{\mathbf{u}}}, \tag{36}$$

$$\mathbf{B}_{\mathrm{POD}\,kl}^{n+1}=\sum_{i=1}^{3\mathscr{N}_{\mathbf{u}}}\sum_{j=1}^{3\mathscr{N}_{\mathbf{u}}}\mathbf{B}_{ij}^{n+1}\mathbf{D}_{ki}^{\mathbf{\Psi}_{\mathbf{u}}}\mathbf{D}_{lj}^{\mathbf{\Psi}_{\mathbf{u}}}, \tag{37}$$

and

$$s_{\mathrm{POD}\,k}^{n+1}=\sum_{i=1}^{3\mathscr{N}_{\mathbf{u}}}s_{i}^{n+1}\mathbf{D}_{ki}^{\mathbf{\Psi}_{\mathbf{u}}}, \tag{38}$$

where $\mathbf{E}_{ij}^{n+1}$ and $\mathbf{B}_{ij}^{n+1}$ are the (i,j)th entry of the matrices $\mathbf{E}^{n+1}$ and $\mathbf{B}^{n+1}$, respectively, which in the colouring scheme can be formed by summation of matrices (see (22)), s_{i}^{n+1} is the ith entry of $\mathbf{s}^{n+1}$, $\mathbf{D}_{ki}^{\mathbf{\Psi}_{\mathbf{u}}}$ and $\mathbf{D}_{lj}^{\mathbf{\Psi}_{\mathbf{u}}}$ are the ith and jth entries of the diagonal matrices $\mathbf{D}_{k}^{\mathbf{\Psi}_{\mathbf{u}}}$ and $\mathbf{D}_{l}^{\mathbf{\Psi}_{\mathbf{u}}}$, respectively,

$$\mathbf{D}_{k}^{\mathbf{\Psi}_{u_x}}=\mathrm{diag}(\mathbf{\Psi}_{u_x k1},\ldots,\mathbf{\Psi}_{u_x k\mathscr{N}_{\mathbf{u}}}), \tag{39}$$

$$\mathbf{D}_{k}^{\mathbf{\Psi}_{u_y}}=\mathrm{diag}(\mathbf{\Psi}_{u_y k1},\ldots,\mathbf{\Psi}_{u_y k\mathscr{N}_{\mathbf{u}}}), \tag{40}$$

$$\mathbf{D}_{k}^{\mathbf{\Psi}_{u_z}}=\mathrm{diag}(\mathbf{\Psi}_{u_z k1},\ldots,\mathbf{\Psi}_{u_z k\mathscr{N}_{\mathbf{u}}}). \tag{41}$$

For quadratic discrete systems and a large reduction of the model space, the matrices $\mathbf{E}_{\mathrm{POD}}$ and $\mathbf{B}_{\mathrm{POD}}$ are most efficiently formed by a summation of matrices in a manner similar to that used above to form the finite element equations using a colouring method. However, due to the global nature of the POD basis functions the number of colors is equal to the number of basis functions. Thus, splitting them into the mean matrices $\overline{\mathbf{E}}_{\mathrm{subPOD}}$, $\overline{\mathbf{B}}_{\mathrm{subPOD}}$ and matrices perturbing the system from these $\mathbf{E}_{\mathrm{subPOD}xk}$, $\mathbf{B}_{\mathrm{subPOD}xk}$ (and similarly for y and z) to obtain:

$$\mathbf{E}_{\mathrm{POD}}^{n+1}=\overline{\mathbf{E}}_{\mathrm{subPOD}}+\sum_{k=1}^{\mathscr{P}_u}\mathbf{E}_{\mathrm{subPOD}xk}a_{u}{}_{x}{}_{k}^{n+1}+\sum_{k=1}^{\mathscr{P}_u}\mathbf{E}_{\mathrm{subPOD}yk}a_{u}{}_{y}{}_{k}^{n+1}+\sum_{k=1}^{\mathscr{P}_u}\mathbf{E}_{\mathrm{subPOD}zk}a_{u}{}_{z}{}_{k}^{n+1}, \tag{42}$$

$$\mathbf{B}_{\mathrm{POD}}^{n+1}=\overline{\mathbf{B}}_{\mathrm{subPOD}}+\sum_{k=1}^{\mathscr{P}_u}\mathbf{B}_{\mathrm{subPOD}xk}a_{u}{}_{x}{}_{k}^{n+1}+\sum_{k=1}^{\mathscr{P}_u}\mathbf{B}_{\mathrm{subPOD}yk}a_{u}{}_{y}{}_{k}^{n+1}+\sum_{k=1}^{\mathscr{P}_u}\mathbf{B}_{\mathrm{subPOD}zk}a_{u}{}_{z}{}_{k}^{n+1}. \tag{43}$$

This decomposition enables any differentiation of these matrices to be easily performed as well as for the reduced model to be rapidly formed every time step.

DOI: 10.1002/fld

5. A PERTURBATION APPROACH FOR FORMING ADJOINT EQUATIONS

This section forms adjoint equations and differentiates mesh/grid-based finite element, spectral element, finite volume, finite difference and reduced models. In general, the global matrix $\mathbf{A}$ (see Equation (4)) is a function of the controls $\mathbf{m}=(m_1\ m_2\ \ldots\ m_{\mathscr{C}})^{\mathrm{T}}$ ($\mathscr{C}$ is the number of controls) and state variables Φ that is $\mathbf{A}=\mathbf{A}(\mathbf{m},\Phi)$. Differentiating (4) with respect to the control variable m_l, the tangent linear model (TLM) can be expressed as

$$\frac{\mathrm{d}\mathbf{A}}{\mathrm{d}m_l}\Phi+\mathbf{A}\frac{\mathrm{d}\Phi}{\mathrm{d}m_l}=\frac{\mathrm{d}\mathbf{s}}{\mathrm{d}m_l}. \tag{44}$$

The gradient of a cost functional J can be expressed as

$$\frac{\mathrm{d}J}{\mathrm{d}m_l}=\left(\frac{\mathrm{d}\Phi}{\mathrm{d}m_l}\right)^{\mathrm{T}}\frac{\partial J}{\partial \Phi}. \tag{45}$$

The TLM (44) can be re-expressed as

$$\mathbf{A}\frac{\mathrm{d}\Phi}{\mathrm{d}m_l}=-\frac{\mathrm{d}\mathbf{A}}{\mathrm{d}m_l}\Phi+\frac{\mathrm{d}\mathbf{s}}{\mathrm{d}m_l}. \tag{46}$$

In addition, it will be convenient to use

$$\frac{\partial \mathbf{s}}{\partial m_l}=\frac{\mathrm{d}\mathbf{s}}{\mathrm{d}m_l}, \tag{47}$$

and

$$\frac{\mathrm{d}\mathbf{A}}{\mathrm{d}m_l}\Phi=\frac{\partial \mathbf{A}}{\partial m_l}\Phi+\mathbf{G}\frac{\mathrm{d}\Phi}{\mathrm{d}m_l}, \tag{48}$$

in which $\mathbf{G}=(\mathbf{g}_1,\mathbf{g}_2,\ldots,\mathbf{g}_{\mathscr{N}})$ where $\mathbf{g}_k=(\partial\mathbf{A}/\partial\phi_k)\Phi$. Also

$$\frac{\mathrm{d}\mathbf{A}}{\mathrm{d}m_l}\Phi=\frac{\partial \mathbf{A}}{\partial m_l}\Phi+\sum_{k=1}^{\mathscr{N}}\frac{\partial \mathbf{A}}{\partial \phi_k}\Phi\frac{\mathrm{d}\phi_k}{\mathrm{d}m_l}, \tag{49}$$

where

$$\sum_{k=1}^{\mathscr{N}}\frac{\partial \mathbf{A}}{\partial \phi_k}\Phi\frac{\mathrm{d}\phi_k}{\mathrm{d}m_l}=\sum_{k=1}^{\mathscr{N}}\mathbf{g}_k\frac{\mathrm{d}\phi_k}{\mathrm{d}m_l}=\mathbf{G}\frac{\mathrm{d}\Phi}{\mathrm{d}m_l}. \tag{50}$$

Therefore, the perturbation of the state variables with respective to the controls can be calculated as

$$\left(\frac{\mathrm{d}\Phi}{\mathrm{d}m_l}\right)^{\mathrm{T}}=\left(-\frac{\partial \mathbf{A}}{\partial m_l}\Phi+\frac{\partial \mathbf{s}}{\partial m_l}\right)^{\mathrm{T}}(\mathbf{A}+\mathbf{G})^{-\mathrm{T}}. \tag{51}$$

Substituting (51) into (45) yields

$$\frac{\mathrm{d}J}{\mathrm{d}m_l}=\left(-\frac{\partial \mathbf{A}}{\partial m_l}\Phi+\frac{\partial \mathbf{s}}{\partial m_l}\right)^{\mathrm{T}}(\mathbf{A}+\mathbf{G})^{-\mathrm{T}}\frac{\partial J}{\partial \Phi}. \tag{52}$$

Defining $(\mathbf{A}+\mathbf{G})^{-\mathrm{T}}\partial J/\partial\Phi=\Phi^*$, the adjoint model is

$$(\mathbf{A}+\mathbf{G})^{\mathrm{T}}\Phi^*=\frac{\partial J}{\partial \Phi}. \tag{53}$$

The gradient of the cost function can, therefore, be calculated using

$$\frac{\mathrm{d}J}{\mathrm{d}m_l}=\left(-\frac{\partial \mathbf{A}}{\partial m_l}\Phi+\frac{\partial \mathbf{s}}{\partial m_l}\right)^{\mathrm{T}}\Phi^*. \tag{54}$$

Notice that the differentiation of the matrix **A** is achieved in a particularly simply way if the matrices are formed for a polynomial system as described in the previous section. Thus for a quadratic system, since the matrix **A** can be represented in the same way as matrix $\mathbf{Q}_{rq}$ in Equation (24),

$$\frac{\partial \mathbf{A}_{ij}}{\partial m_l} = \mathbf{A}^{c}_{uq\,ij} b^{\mathrm{cl}}_{ij}, \tag{55}$$

in which cl is the color of node l and $b^{\mathrm{cl}}_{ij}=1$ if nodes i and j are neighbors to node l; otherwise, $b^{\mathrm{cl}}_{ij}=0$, see Figure 1—far right. The equality becomes an approximation when the discrete matrix **A** has been approximated by a polynomial system. However, even for more complex problems the colouring approach can still be used to simplify the differentiation as described in the next sections.

5.1. Independent Set Perturbation Adjoint (ISP-Adjoint)

The terms $(\partial \mathbf{A}/\partial m_l)\Phi$, $\partial \mathbf{s}/\partial m_l$ in Equation (54) can be determined by hand or AD in the traditional approach to forming discrete or exact gradients of the functional of interest, J. An alternative method for calculating the gradient is to apply a small perturbation, Δm_l, to each of the controls m_l (see Hoffman [39]—one of the first works to form Jacobian's using perturbations). In vector form the perturbation is a vector of length $\mathscr{C}$ with only non-zero at the lth entry, i.e. $\mathbf{\Delta m}_l = (0,\ldots,0,\Delta m_l,0,\ldots,0)^{\mathrm{T}}$. Using this approach the gradients, for small perturbations, are formed using the following approximations:

$$\frac{\partial \mathbf{A}}{\partial m_l}\Phi \approx \frac{\mathbf{A}(\mathbf{m}+\mathbf{\Delta m}_l,\Phi)-\mathbf{A}(\mathbf{m},\Phi)}{\Delta m_l}\Phi = \frac{\mathbf{A}(\mathbf{m}+\mathbf{\Delta m}_l,\Phi)\Phi-\mathbf{A}(\mathbf{m},\Phi)\Phi}{\Delta m_l}, \tag{56}$$

$$\frac{\partial \mathbf{s}}{\partial m_l} \approx \frac{\mathbf{s}(\mathbf{m}+\mathbf{\Delta m}_l)-\mathbf{s}(\mathbf{m})}{\Delta m_l}. \tag{57}$$

Combining these, the key gradient in (54) can be determined:

$$-\frac{\partial \mathbf{A}}{\partial m_l}\Phi + \frac{\partial \mathbf{s}}{\partial m_l} \approx \frac{(-\mathbf{A}(\mathbf{m}+\mathbf{\Delta m}_l,\Phi)\Phi+\mathbf{s}(\mathbf{m}+\mathbf{\Delta m}_l))-(-\mathbf{A}(\mathbf{m},\Phi)\Phi+\mathbf{s}(\mathbf{m}))}{\Delta m_l}. \tag{58}$$

Moreover, perturbations of a number of variables m_l at a given time can be taken without affecting the gradients. However, these must form an independent set of variables, i.e. one variable perturbation should not affect the results of perturbing any other variable. This can be done for the source **s** and for the matrix **A**. In fact, the matrix need not be formed as what is required is matrix vector multiplication:

$$\frac{\mathbf{A}(\mathbf{m}+\mathbf{\Delta m}_l)-\mathbf{A}(\mathbf{m})}{\Delta m_l}\Phi.$$

One can use a higher-order Taylor series expansion than first order to form approximations to

$$-\frac{\partial \mathbf{A}}{\partial m_l}\Phi + \frac{\partial \mathbf{s}}{\partial m_l},$$

from perturbations in m_l, but this requires more computational effort. However, as long as Δm_l is small enough (ignoring round off errors) it will be accurate enough. A good example is the use of $(\mathbf{A}(\mathbf{m}+\mathbf{\Delta m}_l,\Phi)-\mathbf{A}(\mathbf{m}-\mathbf{\Delta m}_l,\Phi))/(2\Delta m_l)$, which will exactly represent a quadratic **A**. It should be noted that for quadratic equations (e.g. Burgers' and the Navier–Stokes equation) and with quadratic discretizations, arbitrary large Δm_l can be taken as the results are independent of the size of Δm_l as **A** and **s** are linear in m.

The r.h.s of the adjoint equation (53) can be formed algebraically (if it has a simple form as it does in the examples presented here) or using the same perturbation approach that is used to differentiate the source, Equation (57). This would enable covariance matrices to be incorporated into the functional J in data assimilation for example.

DOI: 10.1002/fld

The final aspect of the ISP-Adjoint method is the calculation of the matrix **G** used in the adjoint calculation, which is again formed using the independent set colouring approach. Suppose the perturbation vector is $\mathbf{\Delta\Phi}_k = (0,\ldots,0,\Delta\phi_k,0,\ldots,0)^{\mathrm{T}}$ for a perturbation $\Delta\phi_k$ of the kth entry in Φ, then the kth column of **G** is

$$\mathbf{g}_k = \frac{\partial \mathbf{A}}{\partial \phi_k}\Phi \approx \frac{\mathbf{A}(\mathbf{m},\Phi+\mathbf{\Delta\Phi}_k)-\mathbf{A}(\mathbf{m},\Phi)}{\Delta\phi_k}, \quad \Phi = \frac{\mathbf{A}(\mathbf{m},\Phi+\mathbf{\Delta\Phi}_k)\Phi-\mathbf{A}(\mathbf{m},\Phi)\Phi}{\Delta\phi_k}. \tag{59}$$

Notice that this equation has similar form to (56) and thus the same algorithm, and therefore same source code may be used to evaluate both. Suppose all perturbations are equal to $\Delta\phi_k = \varepsilon$, then

$$\mathbf{G} = \frac{1}{\varepsilon}(\mathbf{G}' - \overline{\mathbf{G}}), \tag{60}$$

where $\overline{\mathbf{G}} = (\mathbf{A}\Phi, \mathbf{A}\Phi, \ldots, \mathbf{A}\Phi)$ and $\mathbf{G}' = (\mathbf{g}'_1, \mathbf{g}'_2, \ldots, \mathbf{g}'_{\mathscr{N}\cdot\mathscr{N}_t})$ with

$$\mathbf{g}'_k = \mathbf{A}(\mathbf{m},\Phi+\mathbf{\Delta\Phi}_k)\Phi. \tag{61}$$

In an implementation of the formation of the matrix **G**, one needs to perform matrix vector multiplications involving $\mathscr{N}_c$ vectors. Thus, the entire matrix $\mathbf{G}'$ can be stored within $\mathscr{N}_c$ vectors of length $\mathscr{N}\cdot\mathscr{N}_t$.

Increasing the efficiency of the implementation of the ISP-Adjoint method can be achieved by taking one or more of the following steps:

- Using a discretization or assembly subroutine that does not form the matrices but simply performs matrix vector multiplication while assembling the equations.
- Enabling this subroutine to perform a matrix vector multiplication with a number of solution perturbations simultaneously; thus, to form **G** for finite element discretizations one only needs to integrate across each element once.
- Reducing the number of independent sets or colors by introducing more variables in the solution Φ. For example, by severing the link between elements in continuous (C^1 continuity) finite element discretizations and perturbing this new system. We refer to this method as the Reduced ISP-Adjoint or RISP-Adjoint, which, to round off error, produces identical results to the ISP-Adjoint method. Moreover, a general finite element code, for example, requires no modification to the assembly subroutines since a different linked list for the global solution variables associated with each element is passed to the se subroutines.
- Performing the matrix vector multiplication $\mathbf{G}^{\mathrm{T}}\Phi^*$ without storing matrix $\mathbf{G}^{\mathrm{T}}$.

5.2. *Time dependence and non-linear iteration*

The last point in the efficiency list above can be realized because in time marching methods, typically all of the adjoint solution operates in a lower block structure in which each block is associated with a solution variable at a particular time step. This means that one can solve the adjoint equations by marching backwards in time and the matrix **G** has zeros on the block diagonals of the matrix. Thus, matrix vector multiplication, $\mathbf{G}^{\mathrm{T}}\Phi^*$, can be placed on the r.h.s of the matrix solution for the adjoint in Equation (53). In this case, the matrix $\mathbf{G}^{\mathrm{T}}$ need not be formed and the matrix vector multiplication $\mathbf{G}^{\mathrm{T}}\Phi^*$ involving $\mathbf{G}'^{\mathrm{T}}\Phi^*$ can be made highly efficient for the kth row using

$$(\mathbf{G}'^{\mathrm{T}}\Phi^*)_k = \mathbf{g}_k^{\prime T}\Phi^* = \Phi^{*\mathrm{T}}(\mathbf{A}(\mathbf{m},\Phi\mathbf{\Delta\Phi}_k)\Phi). \tag{62}$$

The structure of the matrix $\mathbf{G}$ in time for the two-level time marching methods is of the general form:

$$\mathbf{G}=\begin{pmatrix} \mathbf{L}^1 & & & \\ \mathbf{K}^2 & \mathbf{L}^2 & & \\ & \ddots & \ddots & \\ & & \mathbf{K}^{\mathcal{N}_t} & \mathbf{L}^{\mathcal{N}_t} \end{pmatrix}. \tag{63}$$

If, instead of linearizing the previous time level value of the solution, one has to iterate to convergence within a time step in order to calculate the non-linear terms at the future time level then the matrices $\mathbf{L}^n$ appearing on the block diagonal of the system of equations occupying the same place as $\mathbf{P}^n$ in the system of adjoint equations would be non-zero. The use of these submatrices $\mathbf{L}^n$ on the block diagonal of $\mathbf{G}$ distinguishes the current approach from that described in [40, 41] for steady-state problems in which a second-order perturbation approach to forming adjoint equations, similar to that used here, is applied using a novel complex variable formulation. Within [40, 41] they maintain the explicit off block diagonal nature of the $\mathbf{G}^n$ matrices (that is $\mathbf{L}^n=0$), which means that the adjoint solution has to follow backwards the non-linear iteration trajectory through the iterative process, which is expensive in terms of memory and computation. Placing non-zero $\mathbf{L}^n$ matrices on the block diagonals of $\mathbf{G}$ means that for steady-state problems, in which non-linear convergence has been achieved, a single linear steady-state adjoint solution is all that is required to form the sensitivities. A second aspect that distinguishes the current approach from that of [40, 41] as well as others is that colouring is not applied to the full sparse Jacobian $(\mathbf{A}+\mathbf{G})^{\mathrm{T}}$ but only to $\mathbf{G}$. $\mathbf{G}$ may have a smaller stencil than $(\mathbf{A}+\mathbf{G})^{\mathrm{T}}$ and therefore needs fewer or equal number of colors than $(\mathbf{A}+\mathbf{G})^{\mathrm{T}}$, which can improve the computational efficiency. For example, the central difference scheme (Equation 6) has matrices $\mathbf{P}^n$ (used to form matrix $\mathbf{A}$) with three non-zeros per row, yet $\mathbf{K}^n$ (used to form $\mathbf{G}$) only has one non-zero per row.

5.3. Simultaneously performing perturbations using graph-colouring methods

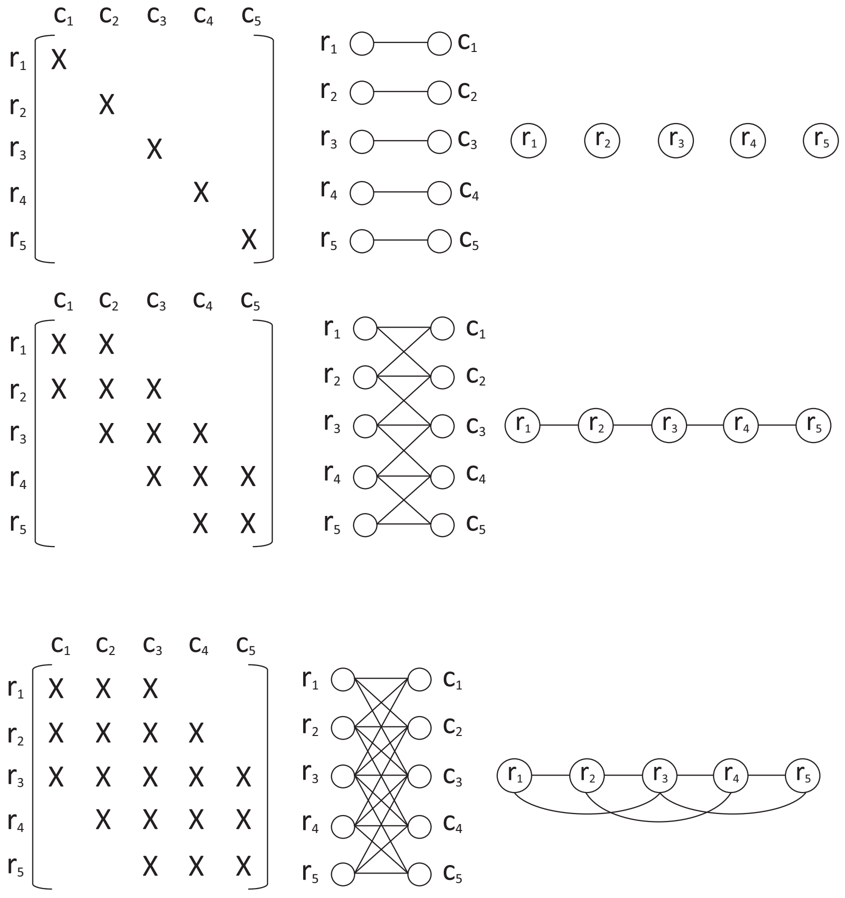
In practice, the perturbations associated with each node or variable k say are grouped in terms of colors and in this way a number of these matrix vector multiplications in (62) can be calculated concurrently, i.e. form the dot product of $\hat{\mathbf{g}}_c'$ with the part of Φ^* associated with node k and form $(\mathbf{G}'^{\mathrm{T}}\Phi^*)_k$ for each row k and for colour c in which $\hat{\mathbf{g}}_c'=\sum_{k\in\text{colour } c}\mathbf{g}_k'$.

The sparse Jacobian matrix $\partial(\mathbf{A}\Phi-\mathbf{s})/\partial\Phi=(\mathbf{A}+\mathbf{G})^{\mathrm{T}}$ can be efficiently computed using graph or matrix colouring. Since the matrix $\mathbf{A}$ is known, the remaining part of this is the formation of the matrix $\mathbf{G}$ (Equation 48) and in particular the formation of the part of $\mathbf{G}$ associated with a given time level n say, i.e. $\mathbf{K}^n$ (similar issues will apply to $\mathbf{L}^n$ if it is no-zero). Once the $\mathbf{K}^n$ sparsity pattern is determined, $\mathbf{K}^n$ and therefore $\mathbf{G}$ and the sparse Jacobian can be computed efficiently. Curtis *et al.* [42] were the first to observe that an orthogonal partition of a Jacobian matrix partition of its columns in which no two columns in a group share a non-zero at the same row index could be used to efficiently determine sparse Jacobian matrices. An orthogonal partition of a Jacobian (or $\mathbf{K}^n$ in this case) can be represented as a distance-2 coloring of the graph representation of the structure of the matrix $\mathbf{K}^n$. Two column vertices that receive the same color in a distance-2 colouring are at a distance greater than two edges from each other (in the graph associated with $\mathbf{K}^n$) and hence are orthogonal. Thus, a distance-2 colouring is a partitioning of the columns of the matrix $\mathbf{K}^n$ into groups of orthogonal columns.

As an example of the structure of the matrix $\mathbf{K}^n$ for a single time step, the sparsity and corresponding graphs for three cases each with five CV cells are shown in Figure 2. These are used by the three schemes that solve Burgers' equation in the applications section. Figure 2 shows from top to bottom the three cases:

(1) A diagonal $\mathbf{K}^n$ matrix (part of the Jacobian matrix) of order 5. The corresponding graph has just vertices (cells) ordered 1 to 5 in 1D with no edges between the vertices although

DOI: 10.1002/fld



$$\begin{array}{c|ccccc} & c_1 & c_2 & c_3 & c_4 & c_5 \\ \hline r_1 & X & & & & \\ r_2 & & X & & & \\ r_3 & & & X & & \\ r_4 & & & & X & \\ r_5 & & & & & X \end{array}$$

$$\begin{array}{c|ccccc} & c_1 & c_2 & c_3 & c_4 & c_5 \\ \hline r_1 & X & X & & & \\ r_2 & X & X & X & & \\ r_3 & & X & X & X & \\ r_4 & & & X & X & X \\ r_5 & & & & X & X \end{array}$$

$$\begin{array}{c|ccccc} & c_1 & c_2 & c_3 & c_4 & c_5 \\ \hline r_1 & X & X & X & & \\ r_2 & X & X & X & X & \\ r_3 & X & X & X & X & X \\ r_4 & & X & X & X & X \\ r_5 & & & X & X & X \end{array}$$

Figure 2. Sparse part of Jacobian matrices $\mathbf{K}^n$ (left panel, X represents the non-zeros in the matrices) and their corresponding bi-partite (middle) and bi-directional (right) graphs. Top panel: a diagonal $\mathbf{K}^n$ matrix is used to determine the sparse Jacobian matrix where one colour is used: rows 1–4 are assigned the first colour; middle panel: a $\mathbf{K}^n$ matrix with 3 non-zeros per row where three colors are used: rows 1 and 4 are assigned the first colour, rows 2 and 5 are assigned the second colour and row 3 is assigned the third colour; bottom panel: a $\mathbf{K}^n$ matrix with 5 non-zeros per row where five colors are used: rows 1, 2, 3, 4 and 5 are assigned one, two, three, four and five colors, respectively.

each vertex is connected to itself (the loop connecting a vertex to itself is not shown in the graph for simplicity).

(2) A $\mathbf{K}^n$ matrix with three non-zeros per row. The graph has links between the cells so that cell i is linked to $i-1$, itself and $i+1$.
(3) A $\mathbf{K}^n$ matrix with five non-zeros per row. The graph has links between the cells so that cell i is linked to $i-2$, $i-1$, itself and $i+1$, $i+2$.

The pictures in Figure 2 of the matrix structures use ‘X’ to mark the potentially non-zeros in the $\mathbf{K}^n$ matrices for the three cases. The graphs have a series of five vertices along a line and have lines (graph edges) below these linking the vertices. Also a line or edge linking each vertex (representing a CV cell and shown on the right) shows the non-zeros in the matrices (on the left). For completeness the bi-partite graph is also shown between bi-directional graph, on the right, and the matrix structure, on the left. It is worth noting that some colouring methods, like the mean field theorem neural network method of obtaining an optimal colouring [43], use matrix vector multiplication as the central part of the approach. This means that matrix equation solver technology (often readily available in FEM and CV codes) may be used to extract parallelization as well as efficiency of the colouring code.

6. SUMMARY OF THE ALGORITHM FOR FORMING THE GRADIENT BASED ON THE ISP-ADJOINT

- Solve Equation (4) for the forward solution Φ;
- Determine the two vectors $\mathbf{A}(\mathbf{m})\Phi$ and $\mathbf{s}(\mathbf{m})$;
- Form the vectors $\mathbf{A}(\Phi, \mathbf{m}+\mathbf{\Delta m}_l)\Phi$ and $\mathbf{s}(\mathbf{m}+\mathbf{\Delta m}_l)$ for all variables l with colour c simultaneously, and repeat $\forall c$ such that $1 \leqslant c \leqslant \mathcal{N}_c$;
- Form the vectors $\mathbf{A}(\Phi+\mathbf{\Delta}\Phi_l, \mathbf{m})\Phi$ for all variables l with colour c simultaneously, and repeat $\forall c$ such that $1 \leqslant c \leqslant \mathcal{N}_c$;
- Form the matrix $\mathbf{G}$ in Equation (59) from the vectors calculated in the previous step;
- Solve adjoint equations (53): $(\mathbf{A}+\mathbf{G})^{\mathrm{T}}\Phi^* = \partial J/\partial\Phi$ for the adjoint Φ^*;
- Determine the gradient using Equation (51).

7. ISP-ADJOINT APPLIED TO BURGERS' EQUATION

For the solution of the Burgers' equation, the global matrix $\mathbf{A}$ (see Equation 4) has the structure with the number of time level $\mathcal{N}_t = 3$:

$$\mathbf{A} = \begin{pmatrix} \mathbf{P}^1 & 0 & 0 \\ -\dfrac{1}{\Delta t}\mathbf{I} & \mathbf{P}^2 & 0 \\ 0 & -\dfrac{1}{\Delta t}\mathbf{I} & \mathbf{P}^3 \end{pmatrix}, \tag{64}$$

whereas the matrix $\mathbf{P}^n$ at time level n (here, the number of nodes $\mathcal{N} = 11$) is

$$\mathbf{P}^n = \begin{pmatrix} \mathbf{P}^n_{1\ 1} & \mathbf{P}^n_{1\ 2} & & & & & \\ \mathbf{P}^n_{2\ 1} & \mathbf{P}^n_{2\ 2} & \mathbf{P}^n_{2\ 3} & & & & \\ & \ddots & \ddots & \ddots & & & \\ & & \mathbf{P}^n_{i\ i-1} & \mathbf{P}^n_{i\ i} & \mathbf{P}^n_{i\ i+1} & & \\ & & & \ddots & \ddots & \ddots & \\ & & & & \mathbf{P}^n_{10\ 9} & \mathbf{P}^n_{10\ 10} & \mathbf{P}^n_{10\ 11} \\ & & & & & \mathbf{P}^n_{11\ 10} & \mathbf{P}^n_{11\ 11} \end{pmatrix}. \tag{65}$$

Using the upwind scheme (7) as an example, the matrix contributions are

$$\begin{aligned} \mathbf{P}^n_{i\ i-1} &= -\max\left(0, \frac{1}{2\Delta x}(u^n_i + u^n_{i-1})\right) - v\frac{1}{(\Delta x)^2}, \\ \mathbf{P}^n_{i\ i} &= \frac{1}{\Delta t} + \max\left(0, \frac{1}{2\Delta x}(u^n_{i+1}+u^n_i)\right) - \min\left(0, \frac{1}{2\Delta x}(u^n_{i+1}+u^n_i)\right) \\ &\quad - \frac{u^n_{i+1} - u^n_{i-1}}{2\Delta x} + 2v\frac{1}{(\Delta x)^2}, \\ \mathbf{P}^n_{i\ i+1} &= \min\left(0, \frac{1}{2\Delta x}(u^n_{i+1}+u^n_i)\right) - v\frac{1}{(\Delta x)^2}. \end{aligned} \tag{66}$$

The matrix $\mathbf{G}$ used in the adjoint calculation has the structure

$$\mathbf{G} = \begin{pmatrix} 0 & 0 & 0 \\ \mathbf{K}^2 & 0 & 0 \\ 0 & \mathbf{K}^3 & 0 \end{pmatrix} \tag{67}$$

DOI: 10.1002/fld

Thus, the matrix equation (53) can be solved by time marching backwards. For all the discretization of Burgers' equation, $\mathbf{K}^n$ is sparse with potentially non-zero values of $\mathbf{K}^n_{i\,i-2}$,$\mathbf{K}^n_{i\,i-1}$,$\mathbf{K}^n_{i\,i}$, $\mathbf{K}^n_{i\,i+1}$,$\mathbf{K}^n_{i\,i+2}$ taken from the coefficients of u^n_{i-2}, u^n_{i-1}, u^n_i, u^n_{i+1}, u^n_{i+2}, respectively, of the advection terms in Equations (6), (7) and (8). The central and upwind discretizations in addition have only the diagonal $\mathbf{K}^n_{ii}$ and tri-diagonals $\mathbf{K}^n_{i\,i-1}$,$\mathbf{K}^n_{i\,i}$, $\mathbf{K}^n_{i\,i+1}$ being potentially non-zero, respectively. Moreover, the point of the ISP-Adjoint method is that the $\mathbf{K}^n$ are formed using a perturbation and colouring method.

The upwind difference scheme (7) has a lower bi-diagonal matrices $\mathbf{K}^n$ for positive advection that is when $u^n_{i+1/2}>0$ (assuming the cells are ordered consecutively from left to right) and two colors ($\mathcal{N}_c=2$) are needed to form the matrices $\mathbf{K}^n$. When there is a mixed sign , the matrix has a bandwidth of three with a maximum of three non-zeros per row and in this case three colors ($\mathcal{N}_c=3$) are needed. Owing to the use of the far field upwind values in the flux limiting calculation of the high-resolution method (8), its matrices $\mathbf{K}^n$ have a bandwidth of five with five non-zeros per row and thus five colors ($\mathcal{N}_c=5$) are necessary and are used here to form the matrices $\mathbf{K}^n$.

The problem solved here has a unit 1D domain and zero boundary conditions at both ends and space and time steps of $\Delta t=0.01$, $\Delta x=0.1$ (11 cells) and a functional $J=\frac{1}{2}\Delta x\sum_{i=1}^{\mathcal{N}}(u_i^{\mathcal{N}_t})^2$ whose sensitivity with respect to the initial conditions ($u_j^0=\exp(-(x_j-0.5)^2/0.1^2)$ for all CV center positions $x_j\in[0,1]$, see Figure 3(a)) is sought and x is the 1D spatial variable. The number of time steps for the 11 cell mesh is $\mathcal{N}_t=50$. As with all calculations presented in this paper, they are performed in double precision unless otherwise stated. The value of $\Delta u=10^{-6}$ is used in the ISP-Adjoint sensitivity calculation. There is also a fine mesh with 100 cells and $\Delta t=0.001$, $\Delta x=0.01$, $\mathcal{N}_t=500$. The initial and final solutions at the final time level are shown in Figures 3(a) and (b) along with the adjoint at the start of time $n=1$ in Figure 3(c) and the sensitivity $\partial J/\partial u_i^0$

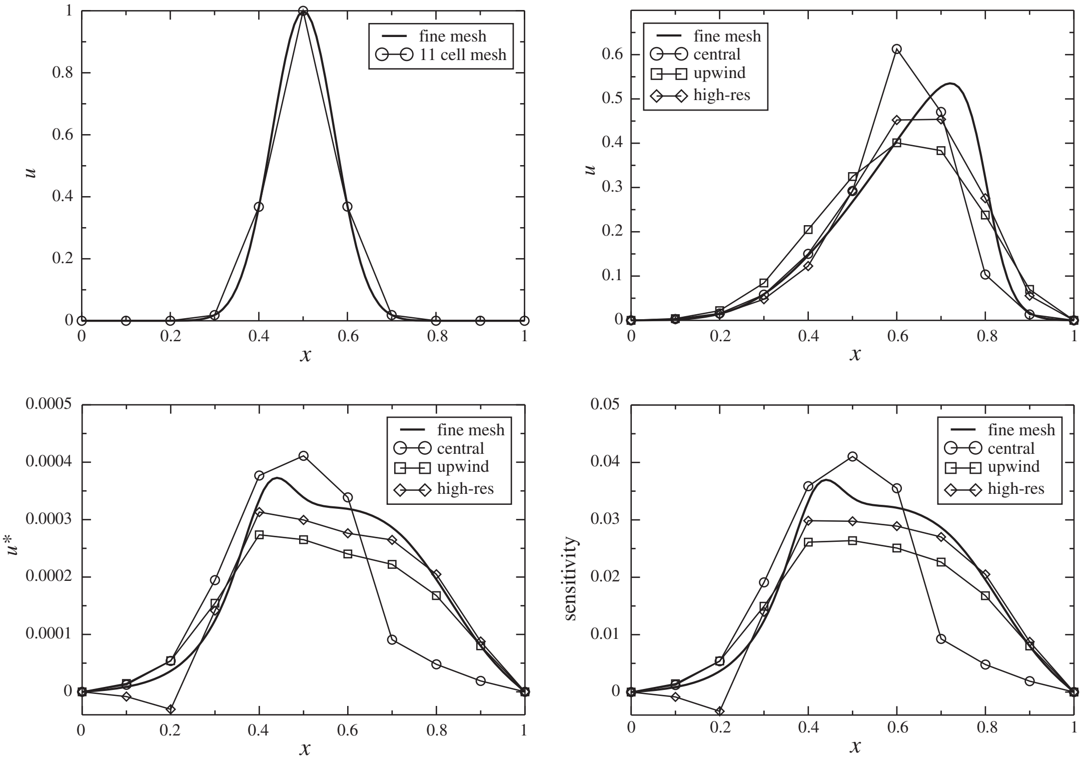


Figure 3. Burgers' equation solution. Top-left: initial conditions for a coarse (11 cells) and fine (101 cells) mesh. Top-right: the forward solution at the end of time $t=0.5$ and for the central difference method, the upwind method and the high-resolution methods along with a fine mesh high-resolution result. Bottom-left: corresponding adjoint and bottom-right: sensitivities for the four simulations shown in top-right.

DOI: 10.1002/fld

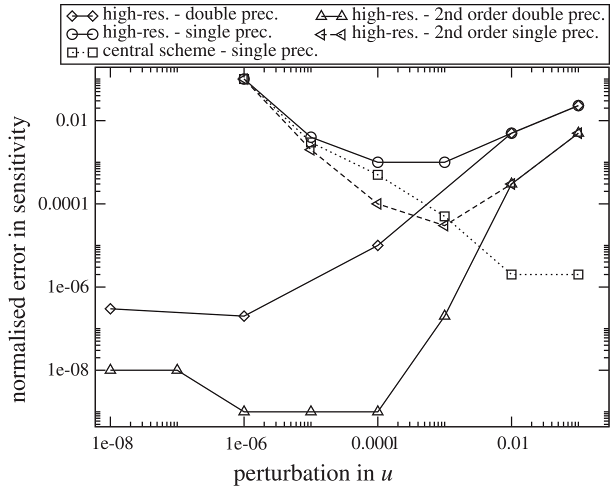


Figure 4. The normalized maximum error in the sensitivity for the coarse mesh calculation against varying Δu in the ISP-Adjoint calculation. The error is normalized with the maximum magnitude of the sensitivity/gradient and the error is determined for both single and double precision. The central scheme has an exact gradient (ignoring round off error) and thus its minimum value yields the best error achievable using single precision. Also shown are the normalized errors in the high-resolution gradients for single and double precisions when a second-order finite difference method is used—that is $(\mathbf{A}(m+\Delta m_l)-\mathbf{A}(m-\Delta m_l))/(2\Delta m_l)$, $(\mathbf{A}(m+\Delta\phi_k)-\mathbf{A}(m-\Delta\phi_k))/(2\Delta\phi_k)$ instead of $(\mathbf{A}(m+\Delta m_l)-\mathbf{A}(m))/\Delta m_l$, $(\mathbf{A}(m+\Delta\phi_k)-\mathbf{A}(m))/\Delta\phi_k$.

in Figure 3(d). The adjoint solution advects from its source,

$$\frac{\partial J}{\partial u_i^{\mathscr{N}_t}} = \Delta x u_i^{\mathscr{N}_t},$$

(which mirrors the forward solution at adjoint time level $\mathscr{N}_t$) backwards through the domain. This propagates the kinetic energy information at the final time level to the initial conditions where the importance of the initial conditions contribution to J can be determined.

It should be noted that the accuracy of the sensitivity/gradient is such that no difference can be observed visually between the exact gradient (obtained by taking perturbations of the individual variables in the initial conditions with a perturbation of 10^{-10}) and the gradients shown in Figure 3(d). The maximum error in the sensitivity $\partial J/\partial u_i^0$, which is normalized by the maximum value of $\partial J/\partial u_k^0, \forall k$, is shown for the high-resolution method versus the perturbation Δu used in the ISP-Adjoint calculation in Figure 4. The sensitivity calculated by the ISP-Adjoint becomes more accurate as Δu is reduced until it gets to the point where round off error interferes with its accuracy and its rate of decrease slows and eventually the error starts to increase. It should be borne in mind that due to the highly non-linear nature of the high-resolution method, the matrix $\mathbf{A}$ is not linear in $u^n \,\forall n$ and thus the perturbation used in the ISP-Adjoint calculation is an approximation. The central difference method (6) has a linear $\mathbf{A}$ in $u^n \,\forall n$ and thus the ISP-Adjoint is exact to computer round off error. In addition, the first-order upwind scheme in (7) is 'mostly' exact. It is only non-exact and non-linear when a face value of the advection velocity (used to advect the velocity u^{n+1}) switches sign on application of the perturbation Δu, which is not very likely to happen at least for small Δu relative to the size of u. In addition, when it does happen the advection velocity will be small and will thus contribute little to the gradient calculation.

The accuracy of the adjoint code can be ascertained through a gradient test [44], where the quantity $w(\alpha)$ is calculated and it is verified that it satisfies $w(\alpha)=1+O(\alpha)$. The quantity w is defined as

$$w(a) = \frac{J(\mathbf{u}^0 + a\mathbf{h}) - J(\mathbf{u}^0)}{a\mathbf{h}^{\mathrm{T}}\nabla J(\mathbf{u}^0)}, \tag{68}$$

DOI: 10.1002/fld

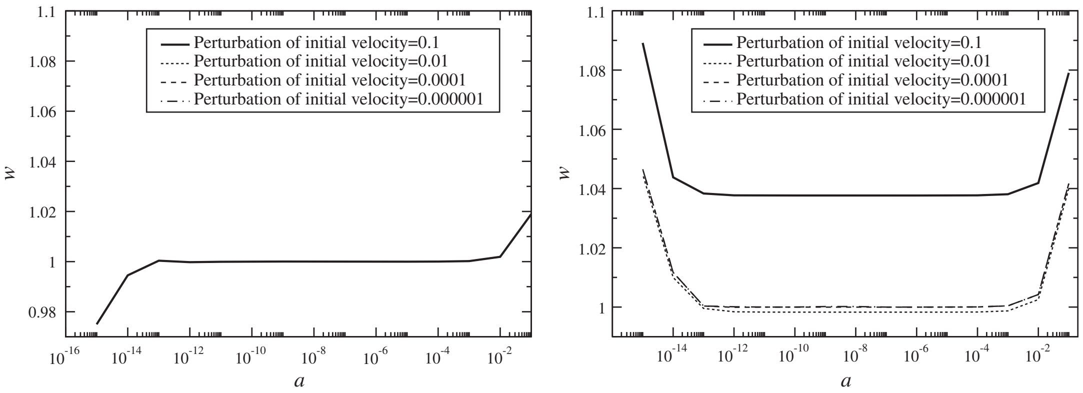


Figure 5. The accuracy of the adjoint model. Left: first-order unwinding; right: flux limiting. Equation (68): $w(a)=(J(\mathbf{u}^0+a\mathbf{h})-J(\mathbf{u}^0))/a\mathbf{h}^{\mathrm{T}}\nabla J(\mathbf{u}^0)$ (here $\mathbf{h}=\nabla J/\|\nabla J\|_2$) is used. The cost function J is defined as the kinematic energy at $t=0.5$ and the control variables are the initial velocities $\mathbf{u}^0$.

where the cost function $J=\frac{1}{2}\Delta x\sum_{i=1}^{\mathcal{N}}(u_i^{\mathcal{N}_t})^2$, $\nabla J(\mathbf{u}^0)=\partial J(\mathbf{u}^0)/\partial\mathbf{u}^0$ and $\mathbf{h}$ is an arbitrary vector of unit length (here $\mathbf{h}=\nabla J/\|\nabla J\|_2$). The vector $\mathbf{u}^0=(u_1^0,u_2^0,\ldots,u_{\mathcal{N}}^0)^{\mathrm{T}}$ of initial CV values of velocity represents the control variables. Here, the initial velocity $u^0=\sum_{j=1}^{\mathcal{N}}M_j u_j^0$ with basis function M_j unity in j and zero elsewhere. For values of a that are small, but not too close to the machine zero, it is expected that $w(a)$ should be close to unity. To quantify the gradient accuracy, Figure 5 shows the variation of the function w with respect to a. It can be seen that, as required, the function $w(a)$ is close to unity when a varies between 10^{-3} and 10^{-12}. Notice that the gradient of the first-order upwind scheme seems independent of the perturbation, which is because it is 'mostly' independent of the size of the perturbation Δu. This property may also be shared with other schemes that have a switch in the discretization depending on if information is going into or out of a cell or element, e.g. Discontinuous Galerkin methods, see [45]. It can thus be concluded, at least for this model problem set-up, that the adjoint model developed here is correct and accurate.

8. ISP-ADJOINT APPLIED TO THE MUNK GYRE

The approach for sensitivity analysis is applied to an idealized ocean gyre problem discretized with a reduced-order model. The sensitivity of the kinetic energy at the end of the simulation with respect to the initial conditions will be sought. Although this is a relatively easily differentiated problem because the discrete system of reduced equations is quadratic and the matrices that change every time step are given by Equations (42) and (43), it does demonstrate how easily applied the ISP-Adjoint algorithm is.

8.1. Discrete adjoint model

For the Navier–Stokes equations, the adjoint model of (4) is written as

$$(\mathbf{A}^{\mathrm{T}}+\mathbf{G}^{\mathrm{T}})\mathbf{\Phi}^*=\left(\begin{pmatrix}\mathbf{P}^{1^{\mathrm{T}}} & \mathbf{H}^{2^{\mathrm{T}}} & & \\ & \mathbf{P}^{2^{\mathrm{T}}} & \ddots & \\ & & \ddots & \mathbf{H}^{\mathcal{N}_t{}^{\mathrm{T}}} \\ & & & \mathbf{P}^{\mathcal{N}_t{}^{\mathrm{T}}}\end{pmatrix}+\mathbf{G}^{\mathrm{T}}\right)\begin{pmatrix}\Phi^{*1}\\ \Phi^{*2}\\ \vdots\\ \Phi^{*\mathcal{N}_t}\end{pmatrix}=\begin{pmatrix}\dfrac{\partial J}{\partial\Phi^1}\\ \dfrac{\partial J}{\partial\Phi^2}\\ \vdots\\ \dfrac{\partial J}{\partial\Phi^{\mathcal{N}_t}}\end{pmatrix}, \tag{69}$$

in which the adjoint solution is $\Phi^{*n}=(\mathbf{u}^{*n^{\mathrm{T}}},\mathbf{p}^{*n^{\mathrm{T}}})^{\mathrm{T}}$.

DOI: 10.1002/fld

8.2. Application to the Munk gyre

The ISP-Adjoint approach has been applied to 2D gyre flows in a computational domain, 1000 km by 1000 km with a depth of $H=500\,\mathrm{m}$. The underlying model equations consist of the 3D incompressible Navier–Stokes equations (for 2D flow cases, one element in the vertical). The wind forcing on the free surface is given by

$$\tau_y=\tau_0\cos(\pi y/L), \quad \tau_x=0.0, \tag{70}$$

where τ_x and τ_y are the wind stresses on the free surface along the x and y directions, respectively, and $L=1000\,\mathrm{km}$. A maximum zonal wind stress of $\tau_0=0.1\,\mathrm{N\,m^{-1}}$ is applied in the latitudinal (y) direction. The Coriolis terms are taken into account using the beta-plane approximation ($f=\beta y$), where $\beta=1.8\times10^{-11}$ and the reference density is $\rho_0=1000\,\mathrm{kg\,m^{-1}}$.

The problem is non-dimensionalized with the maximum Sverdrup balance velocity

$$\beta H\rho_0 v=\frac{\partial\tau}{\partial y}\leqslant\frac{\tau_0\pi}{L}\Rightarrow v\leqslant 3.5\times10^{-2}\,\mathrm{m\,s^{-1}}, \tag{71}$$

(and so the velocity scale $U=3.5\times10^{-2}\,\mathrm{m\,s^{-1}}$ is used here) and the length scale is $L=1000\,\mathrm{km}$. Time is non-dimensionalized with $T=L/U$. The spin-up period is 0.1512 (50 days). The equilibrium state at 50 days is taken as the initial state for both the full and the reduced models. The snapshots are collected from the results obtained in the full model during the simulation period [50,150] days. The time step is 3.78×10^{-4}, equivalent to 3 h. Incorporating the beta-plane approximation yields a non-dimensional $\beta^*=L^2\beta/U=514.3$. The non-dimensional wind stress (applied as a body force here averaged over the depth of the domain) takes the same cosine of latitude profile with $\tau_0^*=\tau_0 L/(U^2\rho_0 H)=163.3$. No-slip boundary conditions are applied to the lateral boundaries. The Reynolds number is defined as $Re=UL/v$ (here the kinematic viscosity is $140\,\mathrm{m^2\,s^{-1}}$). It should be pointed out that since the discrete system of equations is quadratic, application of the ISP-Adjoint is exact (to computational round off) for both the reduced order and the full system of finite element equations.

The POD bases are constructed by the snapshots that are obtained from the numerical solutions by forcing the full forward model with the initial velocity (the background flow). Forty snapshots with 35 POD bases for each component of the velocity field u_x, u_y, u_z and pressure are chosen to capture more than 99.5% of energy (calculated by the first 35 leading eigenvalues). The effect of the number of POD bases on the accuracy of the POD results and the energy percentage captured by the POD bases have been discussed in detail in [38, 46].

In ocean modelling, the pressure term also plays an important role in the geostrophic balance. In this study, taking into account the role of the pressure term in both the POD-Galerkin model and the geostrophic balance, the pressure in the momentum equations is divided into two parts: $p=p_{\mathrm{ng}}+p_{\mathrm{g}}$. To accurately represent geostrophic pressure, its basis functions are split into two sets: $\Phi_{\mathrm{ng}u_x}$ and $\Phi_{\mathrm{ng}u_y}$, which are associated with the u_x- and u_y-velocity components. The geostrophic pressure can be obtained from a quadratic finite element representation while linear finite element representations are used for the velocity components. Furthermore, the geostrophic pressure can be represented by a summation of the two sets of geostrophic basis functions, which are calculated by solving the resulting elliptic equations using a conjugate gradient iterative method (for details see [38]).

In this experiment, the cost function J is defined as the kinematic energy at the final time level ($n=\mathscr{N}_t$ or 200 days):

$$J=\frac{1}{2}\int_\Omega((u_x^{\mathscr{N}_t})^2+(u_y^{\mathscr{N}_t})^2+(u_z^{\mathscr{N}_t})^2)\,\mathrm{d}\Omega. \tag{72}$$

The sensitivity analysis of J with respect to the initial velocity $(u_x^0,u_y^0,u_z^0)^{\mathrm{T}}$ has been carried out, and the corresponding results are shown in Figure 6. The result of applying the ISP-adjoint method to the POD model is similar to that described in Vermeulen and Heemink [47]. This is because in this example the number of colors is equal to the number of POD basis functions used.

DOI: 10.1002/fld

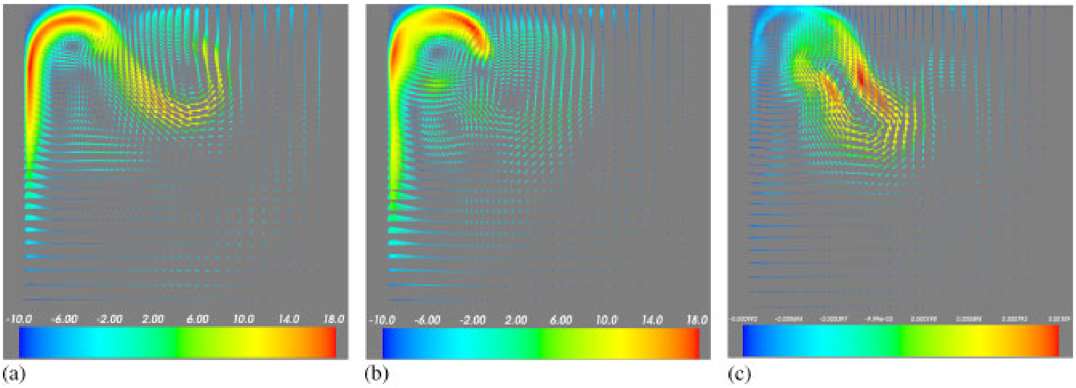


Figure 6. (a) Initial velocity vector ($t = 100$ days); (b) the velocity vector at $t = 200$ days on a regular mesh; and (c) adjoint sensitivity/gradient of the final kinetic energy at $t = 200$ days with respect to the initial velocity.

9. DISCUSSION

This section speculates on the main advantages of the ISP-Adjoint method as well as the problems it is best and worst suited too. Although the method has been applied to fluid-related problems within the current work, it may be applied to any grid-based differential equation, for example finance, solids mechanics or Boltzmann particle transport. The biggest advantages are realized using mesh/grid-based finite element, spectral element, CV or finite difference methods that result in global matrix stencils that have compact support and thus lend themselves to matrix coloring methods in order to extract the efficiencies described here. The approach outlined here would perform poorly for large number of controls or solution variables that have global support, examples of which are provided by full spectral methods of discretizing the differential equations.

The approach may be applied to linear (with or without non-linear discretization methods like flux limiting methods, see application section) or non-linear problems. It may be applied to steady-state or time-dependent problems and is perhaps most simply applied to steady-state problems in which any non-linearities are converged. The latter leads to a rather simple adjoint system of equations that is linear and does not have to follow the trajectory of the forward solution backwards through the non-linear iteration. This advantage is also realized, to some extent, in non-linear time-dependent problems as long as convergence of the solution is realized within the time step. If these solutions are not converged then the only way to form a consistent adjoint is to follow the trajectory of the iteration backwards in the adjoint equations, which can be a lengthy process. Following the forward solution trajectory backwards is effectively what is required, within the adjoint equations, for time-dependent problems and is impossible to avoid. The integration backwards in time aims to obtain, in as efficient manner as possible, the gradient of the cost w.r.t. the control variables. This enables, for example, gradient-based minimization of the functional J.

The ISP-Adjoint method is designed as an alternative to AD methods and thus like AD methods needs further work to deal with situations where the functional that is differentiated has discontinuities (or is non-smooth) for example.

10. CONCLUSIONS

A new method of differentiating potentially complex mesh-based numerical models is presented and applied to simple Burgers' equation and ocean gyre example problems. This method is referred to here as the ISP-Adjoint since differentiation of the matrices and sources of the discrete forward finite element or CV model is realized by colouring variables to form a number of independent sets of variables.

DOI: 10.1002/fld

Three discretization methods namely central difference, first-order upwind and for instance high-resolution schemes were applied to discretize Burgers' equation to demonstrate that exactly the same ISP-Adjoint code can switch between discretizations or parameterizations without changing anything associated with the adjoint or gradient calculations—high-resolution methods are often used for parameterizations or models e.g. for turbulence. A reduced-order Monk gyre problem is solved here to demonstrate the application of the ISP-Adjoint to more complex fluid flow problems.

The ISP-Adjoint method could make more tractable the formation of arbitrarily complex multi-physics mesh-based (4D-VAR) models. It can enable low- and high-order adjoints of models to be manipulated much more freely so that quicker progress can be made in data assimilation, sensitivity and uncertainty analysis, error analysis, adaptive observations and a whole host of other adjoint applications.

ACKNOWLEDGEMENTS

This work was carried out under funding from the U.K.'s Natural Environment Research Council (projects NER/A/S/2003/00595, NE/C52101X/1 and NE/C51829X/1), the Engineering and Physical Sciences Research Council (GR/R60898) and the Leverhulme Trust (F/07058/AB), and with the support from the NURISP framework seven project, the Imperial College High Performance Computing Service and the Grantham Institute for Climate Change. Prof. I. M. Navon would like to acknowledge support of NSF grants ATM-0201808 and CCF-0635162. Many thanks to Tamzin Mole for helping in the technical write up of this work and to Joe Fang for providing the artwork. Finally, the authors would like to thank the two anonymous reviewers who assisted in substantially improving this paper.

REFERENCES

1. Moore AM, Cooper NS, Anderson DLT. Initialisation and data assimilation in models of the Indian Ocean. *Journal of Physical Oceanography* 1987; **17**:1965–1977.
2. Le Dimet FX, Navon IM. Variational and optimization methods in meteorology: a review. *Technical Report*: *Early Review on Variational Data Assimilation*, *SCRI Report No. 144*, 1988; 88. Available from: http://people.scs.fsu.edu/ navon/freqreq.html.
3. Wenzel M, Schröter J, Olbers D. The annual cycle of the global ocean circulation as determined by 4D VAR data assimilation. *Progress in Oceanography* 2001; **48**(1):73–119.
4. Zhu YQ, Navon IM. Impact of parameter estimation on the performance of the FSU global spectral model using its full-physics adjoint. *Monthly Weather Review* 1999; **127**(7):1497–1517.
5. Cacuci DG, Bujor MI, Navon IM. *Sensitivity Uncertainty Analysis*: *Applications to Large-scale Systems*, vol 2. CRC: Boca Raton, 2005.
6. Sanders BF, Katopodes ND. Adjoint sensitivity analysis for shallow water wave control. *Journal of Engineering Mechanics* 2000; **126**:909–919.
7. Daescu DN, Navon IM. Adaptive observations in the context of 4D-Var data assimilation. *Meteorology and Atmospheric Physics* 2004; **85**(4):205–226.
8. Pierce NA, Giles MB. Adjoint recovery of superconvergent functionals from PDE approximations. *SIAM Review* 2000; **42**:247–264.
9. Power PW, Pain CC, Piggott MD, Gorman GJ, Fang F, Marshall DP, Goddard AJH. Adjoint goal-based error norms for adaptive mesh ocean modelling. *Ocean Modelling* 2006; **15**:3–38. DOI: 10.1016/j.ocemod.2006.05.001.
10. Marta AC, Mader CA, Martins JRRA, Van der Weide E, Alonso JJ. A methodology for the development of discrete adjoint solvers using automatic differentiation tools. *International Journal of Computational Fluid Dynamics* 2007; **21**(9–10):307–327.
11. Martins JRRA, Alonso JJ, Reuther JJ. A coupled-adjoint sensitivity analysis method for high-fidelity aero-structural design. *Optimization and Engineering* 2005; **6**(1):33–62.
12. Mohammadi B, Malé JM, Rostaing-Schmidt N. Automatic differentiation in direct and reverse modes: application to optimum shapes design in fluid mechanics. In *Computational Differentiation*: *Techniques*, *Applications*, *and Tools*, Berz M, Bischof CH, Corliss GF, Griewank A (eds). SIAM: Philadelphia, PA, 1996; 309–318.
13. Sherman LL, Taylor AC, Green LL, Newman PA, Hou GW, Korivi VM. First- and second-order aerodynamic sensitivity derivatives via automatic differentiation with incremental iterative methods. *Journal of Computational Physics* 1996; **129**(2):307–331.
14. Griewank A, Walther A. *Evaluating Derivatives Principles and Techniques of Algorithmic Differentiation* (2nd edn). SIAM: Philadelphia, 2008; 438.
15. Boudjemaa R, Cox MG, Forbes AB, Harris PM. Automatic differentiation and its application in metrology. In *Advanced Mathematical and Computational Tools in Metrology VI*, Ciarlini P, Cox MG, Pavese F, Rossi GB (eds). World Scientific Ebook. World Scientific Publishing Co. Pte. Ltd.: Singapore, 2004; 170–179.

16. Coleman TF, Moré JJ. Estimation of sparse Jacobian matrices and graph coloring problems. *SIAM Journal on Numeric Analysis* 1983; **20**:187–209.
17. Coleman TF, Moré JJ. Estimation of sparse Hessian matrices and graph coloring problems. *Mathematical Programming* 1984; **28**:243–270.
18. Coleman TF, Verma A. The efficient computation of sparse Jacobian matrices using automatic differentiation. *SIAM Journal on Scientific Computing* 1998; **19**(4):1210–1233.
19. Coleman TF, Jonsson GF. The efficient computation of structured gradients using automatic differentiation. *SIAM Journal on Scientific Computing* 1999; **20**(4):1430–1437.
20. Hossain S, Steihaug T. Graph coloring in the estimation of sparse derivative matrices: instances and applications. *Discrete Applied Mathematics* 2008; **156**:280–288.
21. Coleman TF, Verma A. Efficient calculation of Jacobian and adjoint vector products in the wave propagational inverse problem using automatic differentiation. *Journal of Computational Physics* 2000; **157**(1):234–255.
22. Gebremedhin AH, Manne F, Pothen A. What color is your Jacobian? Graph coloring for computing derivatives. *SIAM Review* 2005; **47**(4):629–705.
23. Gebremedhin AH. The Enabling Power of Graph Coloring Algorithms in automatic differentiation and parallel processing. In *Dagstuhl Seminar Proceedings 09061*, *Combinatorial Scientific Computing*, Naumann U, Schenk O, Simon HD, Toledo S (eds). LZI: Schloss Dagstuhl, Germany, 2009; ISSN: 1862-4405.
24. Jones MT, Plassmann PE. A parallel graph coloring heuristic. *SIAM Journal on Scientific Computing* 1993; **14**(3):654–669.
25. Baalousha H, Köngeter J. Stochastic modelling and risk analysis of groundwater pollution using form coupled with automatic differentiation. *Advances in Water Resources* 2006; **29**:1815–1832.
26. Glowinski R, Toivanen J. A multigrid preconditioner and automatic differentiation for non-equilibrium radiation diffusion problems. *Journal of Computational Physics* 2005; **207**:354–374.
27. Green LL, Newman PA, Haigler KJ. Sensitivity derivatives for advanced CFD algorithm and viscous modeling parameters via automatic differentiation. *Journal of Computational Physics* 1996; **125**(2):313–324.
28. Bischof CH, Corliss GF, Green L, Griewank A, Haigler K, Newman P. Automatic differentiation of advanced CFD codes for multidisciplinary design. *Computing Systems in Engineering* 1992; **3**(6):625–637.
29. Liu Z, Sandu A. On the properties of discrete adjoints of numerical methods for the advection equation. *International Journal for Numerical Methods in Fluids* 2008; **56**(7):769–803.
30. Reuther J, Alonso JJ, Jameson A, Rimlinger M, Saunders D. Constrained multipoint aerodynamic shape optimization using an adjoint formulation and parallel computers: part i. j. aircraft. *Journal of Aircraft* 1999; **36**(1):51–60.
31. Reuther J, Alonso JJ, Jameson A, Rimlinger M, Saunders D. Constrained multipoint aerodynamic shape optimization using an adjoint formulation and parallel computers: part ii. j. aircraft. *Journal of Aircraft* 1999; **36**(1):61–74.
32. Heimbach P, Hill C, Giering R. An efficient exact adjoint of the parallel MIT general circulation model, generated via automatic differentiation. *Future Generation Computer Systems* 2005; **21**(8):1356–1371.
33. Stammer D, Wunsch C, Giering R, Eckert C, Heimbach P, Marotzke J, Adcroft A, Hill CN, Marshall J. The global ocean circulation during 1992–1997, estimated from ocean observations and a general circulation model. *Journal of Geophysical Research* 2002; **107**(C9):1–27.
34. Piggott MD, Gorman GJ, Pain CC, Allison PA, Candy AS, Martin BT, Wells MR. A new computational framework for multi-scale ocean modelling based on adapting unstructured meshes. *International Journal for Numerical Methods in Fluids* 2008; **56**:1003–1015.
35. Leonard BP. The ULTIMATE conservative difference scheme applied to unsteady one-dimensional advection. *Computing Methods in Applied Mechanics and Engineering* 1991; **88**:17–74.
36. Maffray F. On the coloration of perfect graphs. In *Recent Advances in Algorithms and Combinatorics*, Reed BA, Sales CL (eds). CMS Books in Mathematics, vol. 11. Springer: Berlin, 2003; 65–84. DOI: 10.1007/0-387-22444-0_3.
37. Ford R, Pain CC, Piggott MD, Goddard AJH, de Oliveira CRE, Umbleby AP. A nonhydrostatic finite-element model for three-dimensional stratified oceanic flows. Part I: model formulation. *Monthly Weather Review* 2004; **132**(12):2816–2831.
38. Fang F, Pain CC, Navon IM, Piggott MD, Gorman GJ, Allison PA, Goddard AJH. Reduced order modelling of an adaptive mesh ocean model. *International Journal for Numerical Methods in Fluids* 2009; **59**(8):827–851.
39. Hoffman RN. A four-dimensional analysis exactly satisfying equations of motion. *Monthly Weather Review* 1986; **114**(2):388–397.
40. Nielsen EJ, Kleb WL. Efficient construction of discrete adjoint operators on unstructured grids using complex variables. *AIAA Journal* 2006; **44**(4):827–836.
41. Nielsen EJ, Park MA. Using an adjoint approach to eliminate mesh sensitivities in computational design. *AIAA Journal* 2006; **44**(5):948–953.
42. Curtis A, Powell M, Reid J. On the estimation of sparse Jacobian matrices. *Journal of the Institute of Mathematical Applications* 1974; **13**:117–119.
43. Pain CC, de Oliveira CRE, Goddard AJH. A neural network graph partitioning procedure for grid-based domain decomposition. *International Journal for Numerical Methods in Engineering* 1998; **44**:593–613.
44. Navon IM, Zou X, Derber J, Sela J. Variational data assimilation with an adiabatic version of the NMC Spectral Model. *Monthly Weather Review* 1992; **120**:55–79.

45. Ackroyd RT. Finite element methods for particle transport: applications to reactor and radiation physics. *Research Studies in Particle and Nuclear Technology*, vol. 6. Taylor & Francis Group: London, 1997.

46. Fang F, Pain CC, Navon IM, Piggott MD, Gorman GJ, Allison PA, Goddard AJH. A POD goal-oriented error measure for mesh optimisation. *International Journal for Numerical Methods in Fluids* 2009; DOI: 10.1002/fld.2182.

47. Vermeulen PTM, Heemink AW. Model-reduced variational data assimilation. *Monthly Weather Review* 2006; **134**:2888–2899.