Tangents, adjoints and computational complexity in terrestrial carbon modelling

I. G. Enting[1]

(Received 14 January 2011; revised 6 October 2011)

Abstract

Differentiation enters modelling through initialisation, calibration, sensitivity analysis and data assimilation. Automatic differentiation provides tools for augmenting models to calculate the derivatives. Adjoint transformations lead to computational gains in such analyses. The calculation of tangent models by operator overloading provides a reference case against which to assess such gains. This article uses a vector space representation to analyse how special localisation characteristics of the land surface within the earth system might change the computational complexity of calculating derivatives.

Contents

1 Introduction **C807**

1 Introduction C807

2 Tangent/gradient relations and compilers C808

3 Adjoints C811

4 Terrestrial carbon modelling C816

5 Concluding remarks C818

References C820

1 Introduction

Carbon cycle studies are vital for understanding future global change. Most information is indirect, requiring analysis of inverse problems such as deducing CO_2 fluxes from concentrations [2]. Variational techniques using adjoints have led to a convergence of 'batch' versus 'mass-balance' approaches [8], developing 'process inversions' that estimate parameters in carbon models.

This article considers models defined in terms of differential equations

$$\dot{x}_k = g_k(x_1, \ldots, x_K, t) \quad \text{for } k = 1, 2, \ldots, K\,. \tag{1}$$

While the main model task is integrating such equations, differentiation is important in modelling for sensitivity analysis, calibration, intialisation and data assimilation. Such capabilities turn a numerical 'model' into a 'model analysis system' [7]. Algorithmic differentiation (AD) transforms computer code, using the chain rule to calculate derivatives [5].

As described in Section 2, the calculation of derivatives is simplified if the compiler/language supports operator overloading. Computational complexity is based on matrix descriptions of tangent/adjoint relations to identify efficient uses of AD. Section 3 reviews 'standard' applications of adjoint transformations in vector spaces. Section 4 applies these to the use of AD in terrestrial carbon modelling. The concluding section notes their importance for assessing global change over the 21st century.

2 Tangent/gradient relations and compilers

The starting point is to differentiate equations (1) with respect to a parameter a, as

$$\frac{\partial}{\partial a}\dot{x}_k = \sum_{k'} \frac{\partial}{\partial x_{k'}} g_k(x_1, \ldots, x_K, t) \frac{\partial}{\partial a} x_{k'} + \frac{\partial g_k}{\partial a} \quad \text{for } k = 1, 2, \ldots, K\,. \tag{2}$$

Using the notation v_k to denote the derivatives $\frac{\partial}{\partial a} x_k$, equation (2) is

$$\dot{v}_k - \sum_{k'} v_{k'} h_{k',k}(x_1, \ldots, x_K, t) = b_k(x_1, \ldots, x_K, t)\,. \tag{3}$$

Given $x_k(t)$, equations (3) are linear in the derivatives v_k and are written as $\mathcal{L}\vec{v}(\cdot) = \vec{b}(\cdot)$. This is termed a 'tangent linear model' (TLM). It requires sufficient regularity to allow exchanging derivatives with respect to time and parameters. Derivatives with respect to a set of parameters, $\vec{a}$, with elements a_p, for $p = 1, 2, \ldots, P$, give a family, $v_{kp}(\cdot)$, of solutions. While the 'forcing' b_{kp} depends on p, the TLM 'model', defined by the $h_{k,k'}(\cdot)$, is the same for all p. For simplicity of notation, we consider only linear dependence on $\vec{a}$, represented as a parametric dependence in K basis functions rather than use $K \times P$ functions:

$$\vec{b}(\cdot)[\vec{a}] = \sum_p a_p b_p(\cdot), \quad \vec{v}(\cdot)[\vec{a}] = \sum_p a_p v_p(\cdot)$$

giving the generalisation of (3) as

$$\mathcal{L}\vec{v}(\cdot)[\vec{a}] = \vec{b}(\cdot)[\vec{a}]. \tag{4}$$

Computational complexity is studied for K equations, P parameters and N time steps. The dependence on $[\vec{a}]$ implies a factor of P (or $P+1$ if calculating both $\vec{x}(\cdot)$ and $\vec{v}(\cdot)[\vec{a}]$).

For integration of (1) the operation count grows as ANK where A is the average operation count of evaluating the $g_k(\cdot)$ relative to a simple linear product. It does not affect the analysis if A is a function $A(K)$.

Operator overloading [5, 10] takes a statement operating on real values, for example `S = Q*R` in Fortran denoting $s \leftarrow q \times r$, and re-interprets it by redefining the types of `S`, `Q` and `R` as composite variables. The compiler then interprets `S = Q*R` as also specifying calculation of

$$\frac{\partial s}{\partial a_p} \leftarrow q \times \frac{\partial r}{\partial a_p} + \frac{\partial q}{\partial a_p} \times r \quad \text{for } p = 1, 2, \dots, P\,.$$

More generally, algorithmic differentiation for a binary operator $s \leftarrow f(q, r)$, leads to

$$\frac{\partial s}{\partial a} \leftarrow \frac{\partial f}{\partial q}\frac{\partial q}{\partial a} + \frac{\partial f}{\partial r}\frac{\partial r}{\partial a} \quad \text{for } s \leftarrow f(q, r) \tag{5}$$

with unary operations and functions as special cases. Our Fortran-90 system analysed simple climate models to determine the incremental utility of geosequestration as a 'climate benefit' over time [3]. In principle, modifying models to support AD in this way only requires changes in input, output and type declaration statements. This makes calculation of the TLM by operator overloading an appropriate reference case, with computational complexity $\approx 2ANKP$.

Much of the complexity analysis of linear sensitivity follows from simple properties of matrix/vector multiplication. For dimension $L \times L$, a matrix by vector product requires L^2 multiplications. Direct multiplication of two (square) matrices requires L^3 multiplications. Leaving aside recursive approaches (with $L^{\log_2 7}$ complexity for matrix multiplication), in general a matrix $\mathbf{F}$ (of dimension $L \times L$), factorised as $\mathbf{F} = \prod_{j=1}^{J} \mathbf{F}_j$ and multiplied by a vector, working left to right, $\mathbf{vF}$ takes $(J+1)L^2$ multiplications while $\mathbf{Fv}$ takes

$JL^3 + L^2$ multiplications. Right-to-left evaluation interchanges the operation counts. These operation counts are greatly reduced if the factorisation leads to sparse matrices, $\mathbf{F}_j$.

Tangent/adjoint compilers apply the algorithmic approach of equation (5) to the sensitivities defined by the TLM (2). They analyse the extent to which the model solutions depend on initial conditions, $\vec{x}(0)$, parameters, $\vec{a}$, and model forcing. For generic inputs $\vec{y}$ and generic outputs $\vec{u}$ and related by an operator, $\mathcal{F}$, generally non-linear, as

$$\vec{u} = \mathcal{F}(\vec{y}),$$

sensitivities are expressed as a Jacobian matrix, $\mathbf{J}$, with elements

$$J_{rs} = \frac{\partial u_r}{\partial y_s}\,.$$

Usually, it is neither practical nor interesting to analyse all the sensitivities J_{rs}. A summation $\mathbf{BJC}$, grouping (or selecting) the outputs, z_r, using a matrix $\mathbf{C}$ (a vector if only one output is of interest) and grouping (or selecting) the inputs using a matrix $\mathbf{B}$, gives

$$\mathbf{BJC} = \mathbf{B}\left[\prod_j \mathbf{J}[j]\right]\mathbf{C} \tag{6}$$

with j denoting a time ordering $(\mathbf{J}[1]\mathbf{J}[2]\mathbf{J}[3]\cdots)$ through the calculation. Notionally the dimension of the $\mathbf{J}[j]$ is the number of internal variables in the computer code. From (5), AD corresponds to factorising $\mathbf{J}$ so that all $\mathbf{J}[j]$ have only three elements differing from the identity matrix.

Efficient evaluation of (6) has two special cases each with complexity $2ANK$. If $\mathbf{B}$ reduces to a vector (sensitivities to one input), evaluate left to right ($2PANK$, for $P > 1$). If $\mathbf{C}$ reduces to a vector (sensitivities involving one output), evaluate right to left ($2QANK$ for $Q > 1$). In this second case, the 'integration' runs backwards in time, requiring that the $\vec{x}(\cdot)$ are stored from a forward run, rather than calculated in parallel as in tangent calculations.

Tangent/adjoint compilers transform model code into code for tangents or gradients (derivatives of an output with respect to all inputs) [5, 4]. The following sections use a vector space representation of tangent/gradient relations to clarify cases where special properties might change the relative advantages of tangent versus gradient analysis.

3 Adjoints

Since adjoints are relations for operators on vector spaces, and defined in terms of inner products, these aspects need to be identified. The vector space can be a space of functions (of model solutions or sensitivities), or a discrete data space or a combination of the two. Operators such as $\mathcal{L}$ produce new vectors $\mathcal{A}u$ that are elements of the vector space (with linearity conditions). An inner product, denoted $\langle \cdot \mid \cdot \rangle$, arises when sensitivities are combined or selected using linear relations such as (6). Adjoints are then defined by $\langle v \mid \mathcal{A}u \rangle = \langle \mathcal{A}^{\dagger} v \mid u \rangle$ for all vectors u, v.

Tarantola [11] considered having two vector spaces: a model space (in our terms, a model parameter space) and a data space, with operators mapping between the spaces, inner products denoted $\langle \cdot \mid \cdot \rangle_{\mathrm{M}}$ and $\langle \cdot \mid \cdot \rangle_{\mathrm{P}}$, and the adjoints defined by $\langle v \mid \mathcal{A}u \rangle_{\mathrm{M}} = \langle \mathcal{A}^{\dagger} v \mid u \rangle_{\mathrm{P}}$. This reduces to the definition above by using the sum of the model and data spaces and defining adjoints in this larger space. Similarly, TLM solutions $\vec{v}(\cdot)$ and the forcing $\vec{b}(\cdot)$ may reside in different function spaces, but for simplicity we take these as subspaces of a single larger function space.

For solutions of differential equations to form a vector space usually requires homogeneous boundary conditions. However, various transformations can address this problem, for example by considering the set of solutions as a particular solution plus a vector space of solutions of the homogeneous problem. Fortunately, as noted by Craven [1], in many cases it is not necessary to actually do such a transformation—'it is sufficient to show that it can be done'. Derivatives with respect to parameters will form a vector space if the boundary conditions are independent of the parameters. Cases with parameter dependent boundary conditions can be taken as having parameter independent boundary conditions and a parameter dependent multiple of δ-function forcing at $t = 0^+$.

As noted above, many applications of AD involve 'adjoint modelling', that is, the right to left evaluation of (6). When numerical modellers refer to 'the adjoint' they are usually mean $\mathcal{L}^\dagger$, the adjoint of the TLM, $\mathcal{L}$, (even though $\mathcal{L}$ is equally well 'the adjoint of $\mathcal{L}^\dagger$'). Here $\mathcal{L}$ acts on a set of sensitivities while $\mathcal{L}^\dagger$ acts on a set of integrated weight functions, $\vec{r}(\cdot)$.

Adjoint modelling is of most use when we are interested in some projection of the model solution $\vec{x}(\cdot)$. This is written as an inner product using weighting functions, $w(\cdot)$, as

$$\Theta = \langle \vec{w}(\cdot) \mid \vec{x}(\cdot)[\vec{a}] \rangle. \tag{7}$$

To consider derivatives, we linearise $\vec{x}(\cdot)$ about a reference solution $\vec{x}^*$ as $\vec{x}(\cdot) = \vec{x}^*(\cdot) + \vec{v}(\cdot)[\vec{a}]$. The solution, $\vec{v}(\cdot)[\vec{a}]$, of the linear model (4) is formally represented using a Green's function

$$\vec{v}(\cdot)[\vec{a}] = \mathcal{G}\,\vec{b}(\cdot)[\vec{a}]. \tag{8}$$

The adjoint transformation in modelling separates the operations of differentiation from integration when differentiating expressions such as (7). Formally, with $\vec{r}(\cdot) = \mathcal{G}^\dagger\,\vec{w}(\cdot)$,

$$\begin{aligned}\nabla_{\vec{a}} \langle \vec{w}(\cdot) \mid \vec{v}(\cdot)[\vec{a}] \rangle &= \nabla_{\vec{a}} \langle \vec{w}(\cdot) \mid \mathcal{G}\,\vec{b}(\cdot)[\vec{a}] \rangle \\ &= \nabla_{\vec{a}} \langle \mathcal{G}^\dagger\,\vec{w}(\cdot) \mid \vec{b}(\cdot)[\vec{a}] \rangle \\ &= \nabla_{\vec{a}} \langle \vec{r}(\cdot) \mid \vec{b}(\cdot)[\vec{a}] \rangle. \end{aligned} \tag{9}$$

In matrix terms, the adjoint transformation can be regarded as re-writing the sum

$$\sum_{n',k'} w_{n',k'} \left[\sum_{n,k} G_{n'k',nk} b_{nkp} \right] = \sum_{n,k} \left[\sum_{n',k'} w_{n'k'} G_{n'k',nk} \right] b_{nkp}$$

changing $(NK)^2 + (NK)^2P$ multiplications into $(NK)^2 + NKP$ multiplications. Since the matrix of $G_{n'k',nk}$ is usually unavailable, and N^2 dependence makes either form highly inefficient, most analyses of adjoints use differential operators $\mathcal{L}$ and $\mathcal{L}^\dagger$, rather than integral operators $\mathcal{G}$ and $\mathcal{G}^\dagger$,

$$\begin{aligned} \nabla_{\vec{a}} \langle \vec{w}(\cdot) \mid \vec{v}(\cdot)[\vec{a}] \rangle &= \nabla_{\vec{a}} \langle \mathcal{L}^\dagger \vec{r}(\cdot) \mid \vec{v}(\cdot)[\vec{a}] \rangle \\ &= \nabla_{\vec{a}} \langle \vec{r}(\cdot) \mid \mathcal{L} \vec{v}(\cdot)[\vec{a}] \rangle \\ &= \nabla_{\vec{a}} \langle \vec{r}(\cdot) \mid \vec{b}(\cdot)[\vec{a}] \rangle \end{aligned} \tag{10}$$

with $\vec{w}(\cdot) = \mathcal{L}^\dagger \vec{r}(\cdot)$ giving equations for the adjoint model.

If we consider $\mathcal{G}$ in a block matrix form, partitioned by n, then we have $G_{n'k',nk} = 0$ for $n' < n$ and so the transpose G^T has $G^T_{n'k',nk} = 0$ for $n < n'$ —the dependence evolves backwards in time. Thus, consistent with the evaluation of the factors of $\mathbf{J}$ in time reversed order, backwards evaluation is necessary for any differential relation derived from $\mathcal{G}^\dagger$ (but of course this does not prove that such a form exists).

The notional complexity of using the TLM is $KN + A'NKP$. The KN is for evaluating the inner product and $A'NKP$ is for integrating the TLM for P parameters. In contrast, the complexity of the adjoint approach is $A^\dagger NK + NKP$. The $A^\dagger NK$ is for evaluating the adjoint function and NKP is for evaluating the inner products. In terms of binary operations, we expect $A^\dagger \approx A' \approx 2A$.

This analysis assumes the complexity of calculating $\vec{b}[\vec{a}]$ is less than $A'NKP$. This is true if only a small fraction of A operations in the original models involve parameters and if most parameters are associated with small sets of prognostic variables. Each of the AKN binary operations involves at most two parameters and more usually only one or zero. In some cases, the reduction of NKP components in $\vec{b}[\vec{a}]$ down to NK (or less) is explicit when each p applies to only one value of k as in surface fluxes for CO$_2$ inversions. Thus calculation of $\vec{b}[\vec{a}]$ generally has a complexity $\leqslant AKNP$.

The analysis above complements the description from sparse matrix factorisation. The adjoint is more efficient because the factor $A^\dagger$ (characterising adjoint equations) is separated from the term that includes P (the number of parameters). In operational terms, this separates integration from differentiation. Using the vector space formalism raises the possibility of identifying special cases where further reductions in computational complexity might be possible.

Before looking at the special case of terrestrial carbon modelling, we review various well known generic applications of adjoint modelling using a common framework to analyse (i) sensitivity analysis, (ii) fitting soft constraints, and (iii) optimisation in the presence of hard constraints. For computational complexity, parametric dependence on $\vec{a}$ implies a factor of P in the complexity measure.

Differentiation (as in sensitivity analysis) This uses (9) and (10) to give $\nabla_{\vec{a}}\langle\vec{w}(\cdot) \mid \vec{x}(\cdot)[\vec{a}]\rangle = \nabla_{\vec{a}}\langle\vec{r}(\cdot) \mid \vec{b}(\cdot)[\vec{a}]\rangle$ with $\vec{r}(\cdot) = \mathcal{G}^\dagger\vec{w}(\cdot)$ or $\vec{w}(\cdot) = \mathcal{L}^\dagger\vec{r}(\cdot)$ where $\mathcal{L}$ is the evolution operator for the TLM and $\mathcal{G}$ is its Green's function. For large P using the adjoint becomes more efficient than multiple integrations of tangent linear models.

Gradients for soft constraints When a cost function, Θ, has the form of a squared 'data-mismatch' function, defined by a projection $\vec{H}$ from the function space $\vec{x}(\cdot)$ onto a discrete data set $\vec{z}$, iterative minimisation of Θ with respect to parameters $\vec{a}$ can be facilitated by using derivatives $\nabla_{\vec{a}}\Theta$. Putting

$$\vec{x}(\cdot)[a] \approx \vec{x}(\cdot)[\vec{a}_0] + \vec{v}(\cdot)[\vec{a}'], \tag{11}$$

where $x(\cdot)[\vec{a}_0]$ is current estimate, gives

$$\begin{aligned}
\nabla_{\vec{a}}\langle \vec{H}\,\vec{x}(\cdot)[\vec{a}] - \vec{z} \mid \vec{H}\,\vec{x}(\cdot)[\vec{a}] - \vec{z}\rangle &= 2\nabla_{\vec{a}}\langle \vec{H}\,\vec{x}(\cdot)[\vec{a}_0] - \vec{z} \mid \vec{H}\,\vec{v}(\cdot)[\vec{a}']\rangle \\
&= 2\nabla_{\vec{a}}\langle \vec{H}^\dagger(\vec{H}\,\vec{x}(\cdot)[\vec{a}_0] - \vec{z}) \mid \vec{v}(\cdot)[\vec{a}']\rangle \\
&= 2\nabla_{\vec{a}}\langle \mathcal{L}^\dagger\,\vec{r}(\cdot) \mid \vec{v}(\cdot)[\vec{a}']\rangle \\
&= 2\nabla_{\vec{a}}\langle \vec{r}(\cdot) \mid \mathcal{L}\,\vec{v}(\cdot)[\vec{a}']\rangle \\
&= 2\nabla_{\vec{a}}\langle \vec{r}(\cdot) \mid \vec{b}(\cdot)[\vec{a}]\rangle, \qquad (12)
\end{aligned}$$

where

$$\mathcal{L}^\dagger\vec{r}(\cdot) = \vec{H}^\dagger(\vec{H}\,\vec{x}(\cdot)[\vec{a}_0] - \vec{z}). \qquad (13)$$

The representation (12) uses a composite space—the top line is in the data subspace—the remaining lines are in the function subspace(s). At each step of the iterative minimisation, the derivatives of Θ are defined by the inner product of $\vec{b}(\cdot)$ and the functions $\vec{r}(\cdot)$ obtained by integrating (13). For large numbers of parameters, iterative solutions based on this transformation can be computationally efficient even when $\vec{x}(\cdot)$ depends linearly on $\vec{a}$. Applications in terrestrial carbon modelling include calibration and data assimilation.

Gradients, with hard constraints The problem is to minimise a functional $\Theta[\vec{u}(\cdot)]$, subject to $\vec{u}(\cdot)$ exactly satisfying equations

$$\mathcal{L}\vec{u}(\cdot) = 0 \qquad (14)$$

as a 'hard' constraint. Such problems are solved by introducing a function $\vec{r}(\cdot)$ as a Lagrange multiplier and minimising

$$\Theta^* = \Theta[\vec{u}(\cdot)] - \langle \vec{r}(\cdot) \mid \mathcal{L}\vec{u}(\cdot)\rangle. \qquad (15)$$

The solutions for the constrained minimisation are obtained from

$$\nabla_{\vec{u}}\Theta^* = \nabla_{\vec{u}}\Theta - \nabla_{\vec{u}}\langle \mathcal{L}^\dagger\vec{r}(\cdot) \mid \vec{u}(\cdot)\rangle = 0\,, \qquad (16)$$

giving adjoint equations to define the Lagrange multiplier

$$\mathcal{L}^{\dagger}\vec{r}(\cdot) = \nabla_{\vec{u}}\Theta\,. \tag{17}$$

Solution of (14) and (17) defines the minimum. In carbon cycle studies, problems of deducing sources and sinks can be formulated this way including mass-balance inversions of space-time distributions of $\mathrm{CO_2}$ and estimating global sources from $\mathrm{CO_2}$ in bubbles trapped in ice.

4 Terrestrial carbon modelling

Inverse modelling of the carbon cycle progressed from flux estimation, through to process calibration and explorations of data assimilation [12]. The importance of differentiation in such calculations motivated the present analysis of computational complexity, taking account of the localised nature of the terrestrial system. Terrestrial carbon models usually have a number (K_G) of independent instances of a single model (or a small number of biome specific models), with spatially explicit forcing distinguishing the instances, each with K_R prognostic variables.

The principle underlying 'process inversion' [7] is that a small number of parameters constrain a large set of fluxes. The utility of this approach relies heavily on this assumption. This restriction does not preclude effects such as parameters having a spatial modulation due to adaptation to mean climate, so long as this can be parameterised.

The impact of the terrestrial system on the physical climate involves complex non-linear relations. Some components of an earth system model may be easier than others for implementing adjoints. The following discussion explores whether, for the terrestrial system, having A of the form $A_R \times K_G$ simplifies the various types of application considered above.

Initialisation Initialisation finds a quasi-steady state to start an integration in time—in terrestrial carbon modelling this is usually a seasonally periodic state, forced by representative meteorological fields, requiring state variables that give zero year-to-year change. Initial tests show that AD accelerates the equilibration of models of nutrient limited carbon pools.

Parameter sensitivity As noted above, terrestrial carbon modelling aims to capture disparate fluxes using small a number of parameters that determine the response to disparate meteorological forcing. To capture how sensitive fluxes are to errors and uncertainties in these parameters, direct application of the TLM is appropriate. Characterising sensitivity to errors in meteorological forcing is an important but ill-defined problem.

Bottom-up calibration 'Bottom-up' refers to calculations that assemble a large scale description from detailed local information. In developed nations such information is assembled for various land management purposes. However, the use of remotely sensed data from satellites is the only way of achieving global coverage. Usually this requires calibration against reference sites, applying the formalism of (12) to a cost function

$$\Theta_{\mathrm{BU:cal}} = \sum_{k \subset [1, K_G]} \Theta_k \,. \tag{18}$$

If only a small number of parameters (per biome) is involved, the complexity AK_GK_RNP makes the use of the TLM effective. Otherwise a local adjoint model must be run K_G times, with K_G data mis-matches as 'forcing'.

Top-down calibration 'Top-down' refers to constraints linking multiple locations, for example, the constraints from space-time distributions of CO$_2$. The atmospheric CO$_2$ distribution $\vec{x}_A$ is related to surface fluxes by atmospheric transport $\mathcal{T}\vec{x}_A = \vec{\Phi}$ and $\vec{\Phi}$ related to terrestrial carbon variables $\vec{x}_T$. Since these inversions are ill-conditioned, a regularisation constraint is needed. Spatial smoothing can be obtained from correlation constraints [8] or geostatistical considerations. Alternatively, top-down estimates can be constrained by local 'bottom-up' information with Θ_{BU} from (18) as a Bayesian prior for top-down inversion.

If top-down and bottom-up data are combined by minimising a cost function, $\Theta = \Theta_{\mathrm{TD}} + \Theta_{\mathrm{BU}}$, there is no necessity to use the same transformation for both $\nabla_{\vec{a}}\Theta_{\mathrm{BU}}$ and $\nabla_{\vec{a}}\Theta_{\mathrm{TD}}$. Thus while $\nabla_{\vec{a}}\Theta_{\mathrm{BU}}$ can be conveniently calculated using a tangent form, $\nabla_{\vec{a}}\Theta_{\mathrm{TD}}$ would be most conveniently calculated from the adjoint of the transport model if available. Generally, the k-dependence of $\frac{\partial}{\partial a_p}\Phi_k$ will not match pre-computed regions and in iterative solutions of non-linear problems the k-dependence will change between iterations.

Data assimilation Various forms of data assimilation incorporate observations into an evolving model state, by minimising a composite cost function. In land surface models such tasks could include detection and quantification of carbon climate feedbacks in the medium to long term. Estimates range from 20 to 200 ppm of extra $\mathrm{CO_2}$ expected in the atmosphere this century from such feedbacks. Operational carbon data assimilation can be important in Natural Resource Management [9]. Data assimilation also provides recursive parameter estimation as an alternative to a 'batch' approach of simultaneously fitting all parameters.

5 Concluding remarks

This study was motivated by development of the Australian Community Climate and Earth System Simulator (ACCESS) whose land surface component is based on earlier CSIRO models. This is augmented by carbon pools based on the CASA model [6] which was used in tests of the Fortran-90 AD procedures. Tests with this model confirm that the simplicity of AD by operator overloading applies in practice, even with an existing model not designed for such transformations.

In the earth system, the land surface components, water and carbon, share important characteristics: (i) high spatial and temporal heterogeneity; (ii) multiple time scales; (iii) disparate calibration data; and (iv) primarily local interactions. While many of the characteristics of land surface models make calibration challenging, characteristic (iv) simplifies the calibration calculations, as indicated by the results from the preceding sections. To show this, we review the results from the preceding sections, for sensitivity of Q outputs to P parameters in the case of N time steps, for K variables, and A operations per time step.

From the sparse matrix analysis of AD it follows that: (i) an integration of complexity AKN can have P derivatives calculated with complexity $cAKNP$ using AD implemented by operator overloading (tests give c of about two to three); and (ii) derivatives of Q outputs can be evaluated with complexity $c'AKNQ$ with'gradient' code. This is desirable if P is large and $P \gg Q$. Calibration calculations, having $Q = 1$, are the most important case.

In terrestrial modelling with $K = K_G \times K_R$ (K_G grid cells with K_R reservoirs, or other variables, per cell) the vector space form helps clarify special cases. What is less obvious from the sparse matrix form and what we learn from the vector space form for the case $K = K_G \times K_R + K_T$ is (iii) the balance shifts in favour of using a tangent model for calibrations where the cost function is a sum of K_G local cost functions; (iv) $K = K_G \times K_R + K_T$ with K_T variables describing atmospheric transport of carbon. Calibrations that combine top-down and bottom up criteria are not restricted to making a global choice between 'tangent' and 'gradient' approaches for all aspects of differentiation.

In conclusion, the vector space representation is useful for analysing the computational complexity of a range of modelling calculations in terrestrial carbon studies.

Acknowledgements The ARC funds the Centre of Excellence for Mathematics and Statistics of Complex Systems. The CSIRO supports Ian Enting's fellowship. The ARC Network for Earth System Science supported work on AD by Mark Fielding. ACCESS team members gave valuable feedback. The final version benefited greatly from the careful reading by the anonymous reviewer. YingPing Wang helped test AD in the CASACNP model. This work began at the MSRI Workshop on Carbon Data Assimilation, with hospitality from Prof. Inez Fung.

References

[1] B. D. Craven. *Control and Optimization.* Chapman and Hall, London, 1995. C812

[2] I. G. Enting. *Inverse Problems in Atmospheric Constituent Transport.* CUP, Cambridge, UK, 2002. C807

[3] I. G. Enting, D. M. Etheridge, and M. J. Fielding. A perturbation analysis of the climate benefit of geosequestration. *Int. J. Greenhouse Gas Control*, 2:289–296, 2008. doi:10.1016/j.ijggc.2008.02.005 C809

[4] R. Giering. Tangent linear and adjoint biogeochemical models. In P. Kasibhatla et al., editor, *Inverse Methods in Global Biogeochemical Cycles. (Geophysical Monograph no. 114)*, pages 33–48. AGU, Washington, DC, 2000. C811

[5] A. Griewank. *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation.* SIAM, Philadelphia, 2000. C807, C809, C811

[6] C. S. Potter, J. T. Randerson, C. B. Field, P. A. Matson, P. M. Vitousek, and H. A. Mooney. Terrestrial ecosystem production: A process model based on global satellite and surface data. *Global Biogeochemical Cycles*, 7:811–841, 1993. doi:10.1029/93GB02725 C818

[7] P. J. Rayner, M. Scholze, W. Knorr, T. Kaminski, R. Giering, and H. Widmann. Two decades of terrestrial carbon fluxes from a carbon cycle data assimilation system (CCDAS). *Global Biogeochemical Cycles*, 19:GB2026, 2005. doi:10.1029/2004GB002254 C807, C816

Table 1: notation.

$\vec{a}$	Vector of parameters, a_p, $p = 1, 2, \ldots, P$.
A	Average complexity of evaluating rates of change.
$\mathcal{A}$	Arbitrary operator on a vector space, with adjoint $\mathcal{A}^\dagger$.
$\vec{b}(\cdot)$	Forcing in Tangent Linear Model, $\vec{b}(\cdot)[\vec{a}]$, when projected onto parameter vector $\vec{a}$.
$f(\cdot,\cdot)$	Functional relation specifying a generic binary operation in a numerical calculation.
$g_k(\cdot)$	Functions specifying rates of change of prognostic variables.
$\mathcal{G}$	Green's function operator for tangent linear model.
$h_{k',k}(\cdot)$	Rate coefficients, $\frac{\partial}{\partial x_{k'}} g_k(\cdot)$, in tangent model.
$\vec{H}$	Projection of linearised model onto observation space.
$\mathbf{J}[j]$	Factor in sparse matrix factorisation of Jacobian matrix $\mathbf{J}$, factorised as $\prod_j \mathbf{J}[j]$.
K	Number of differential equations, indexed by k.
K_G	Number of locations at which terrestrial model occurs
K_R	Number of reservoirs (or other prognostic variables) in each occurrence of terrestrial model.
$\mathcal{L}$	Differential operator for tangent linear model.
N	Number of time steps.
t	Time.
$\vec{u}$	Arbitrary vector of model outputs.
$\vec{v}(\cdot)[\vec{a}]$	Linear perturbation of $\vec{x}(\cdot)$
$v_k(\cdot)$	Derivative (with respect to parameter a) of generic variable x_k.
$w(\cdot)$	Weighting function (for example, in the definition of cost function) expressed as abstract vector.
$\vec{x}(\cdot)$	Model solutions as functions of time and with components $x_1(\cdot), \ldots, x_K(\cdot)$.
$\vec{x}^*(\cdot)$	Reference case of $\vec{x}(\cdot)$.
$\vec{y}$	Arbitrary vector of model inputs.
$\vec{z}$	Generic set of observations, $z_1, \ldots, z_Q$.
Θ	Generic cost function.

[8] C. Rödenbeck. Estimating CO_2 sources and sinks from atmospheric mixing ratio measurements using a global inversion of atmospheric transport. Max-Planck-Institut für Biogeochemie: Technical Paper 6, 2005. http://www.bgc-jena.mpg.de/mpg/websiteBiogeochemie/Publikationen/Technical_Reports/tech_report6.pdf C807, C817

[9] W. Steffen, editor. *Blueprint for Australian Carbon Cycle Research.* Australian Greenhouse Office, Canberra, 2005. http://www.globalcarbonproject.org/global/pdf/Australia.CarbonBluePrint_2005.pdf C818

[10] C. W. Straka. ADF95: Tool for automatic differentiation of a FORTRAN code designed for large numbers of independent variables. *Comput. Phys. Commun.*, 168:123–139, 2005. doi:10.1016/j.cpc.2005.01.011 C809

[11] A. Tarantola. *Inverse Problem Theory: Methods for Data Fitting and Model Parameter Estimation.* SIAM, Philadelphia, 2005. C811

[12] Y.-P. Wang, C. M. Trudinger, and I. G. Enting. A review of applications of model-data fusion to studies of terrestrial carbon fluxes at different scales. *Agricultural and Forest Meteorology*, 149:1829–1842, 2009. doi:10.1016/j.agrformet.2009.07.009 C816

Author address

1. **I. G. Enting**, ARC Centre of Excellence for Mathematics and Statistics of Complex Systems, 139 Barry St., University of Melbourne, Victoria 3010, Australia.
mailto:ienting@unimelb.edu.au