Geosci. Model Dev. Discuss., 6, 5251–5288, 2013
www.geosci-model-dev-discuss.net/6/5251/2013/
doi:10.5194/gmdd-6-5251-2013


Geoscientific Model Development Discussions
Open Access



# ADISM v.1.0: an adjoint of a thermomechanical ice-sheet model obtained using an algorithmic differentiation tool

**J. McGovern[1,*], I. Rutt[1], J. Utke[2], and T. Murray[1]**

[1]Department of Geography, College of Science, Swansea University, Singleton Park, Swansea SA2 8PP, UK
[2]Argonne National Laboratory, Argonne, IL, USA
[*]now at: Laboratoire de Glaciologie et Géophysique de l'Environnement, UJF-Grenoble, CNRS – UMR5183, Saint-Martin-d'Hères, France

Received: 26 August 2013 – Accepted: 17 September 2013 – Published: 9 October 2013

Correspondence to: J. McGovern (jonathan.mcgovern@lgge.obs.ujf-grenoble.fr)

Published by Copernicus Publications on behalf of the European Geosciences Union.







## Abstract

A number of problems in contemporary glaciology could benefit from the application of adjoint models. On a simple level, adjoint models can be used to calculate ice-sheet sensitivities with respect to spatially varying parameters such as the basal sliding coefficient. At a more sophisticated level, adjoint models may be used as components of variational data assimilation schemes, allowing problems of model initialization and data-constrained evolution to be tackled.

Fundamentally, adjoint models calculate the sensitivity of a cost function to a suite of control parameters. Such model sensitivities can alternatively be obtained by running the model many times, perturbing each control parameter separately in turn, and calculating the resulting sensitivity in each case. For large numbers of control parameters, however, such as the case where a control parameter corresponds to each point in the model domain, the computational cost becomes prohibitive. The use of adjoint models allows sensitivities to be obtained more efficiently – adjoint model sensitivities are obtained in a single run – and more accurately, since the differentiation of the model is done with machine precision.

We present a finite-difference shallow ice approximation (SIA), thermomechanical ice-sheet model (the forward model), and its adjoint, as generated by using the OpenAD algorithmic differentiation tool. We verify the ice-sheet model using standard SIA benchmark tests and check the consistency between derivatives computed by OpenAD and certain numerically approximated derivatives. Typical adjoint calculations are demonstrated by application to the Greenland ice sheet.

## 1 Introduction

In recent decades, adjoint models have found broad application in the geosciences, perhaps most notably as components of atmospheric data assimilation systems (Lahoz et al., 2010). In atmospheric modeling, it is crucial to obtain an estimation of the





GMDD
6, 5251–5288, 2013



initial state of the atmosphere that is consistent with the model physics; otherwise the model solution will be dominated by fast, transient modes as the model atmosphere adjusts toward balance. In weather forecasting and climate reanalysis applications, the assimilation of data in a given time window is necessary. Both problems can be addressed by using variational assimilation methods, which have adjoint models at their core (Courtier et al., 1993).

Researchers have expressed growing interest in inverse methods in glaciology over the past 20 yr, and consequently adjoint models have been studied in relation to a number of glaciological problems. Perhaps the first instance of a glaciological adjoint model in the literature is due to MacAyeal (1992), who obtained the adjoint of a depth-integrated ice-sheet model analytically and used it to invert basal traction parameters for an Antarctic ice stream. More recently, analytical adjoints have been presented by, among others, Brinkerhoff et al. (2011) and Goldberg and Sergienko (2011): the former use an adjoint to assess the sensitivity of basal conditions under a Greenland outlet glacier to basal traction and geothermal heat flux, while the latter examine nonlinearities in the context of inverse methods.

In addition to their use in data assimilation, adjoint models may be used by themselves to calculate the sensitivity of models to a specified set of parameters. In glaciology, the future of the Greenland and Antarctic ice sheets is of particular interest; the Greenland ice sheet (GrIS), for example, would cause sea level to rise by approximately 7 m if it were to melt completely (Cazenave, 2006). In trying to understand the likely mass loss from the GrIS over the coming centuries, it would be helpful to understand the sensitivity of the ice sheet to various spatially varying forcing fields. Ordinarily, model sensitivities to such spatially varying parameters are obtained by perturbing the parameter at each model grid point in turn and calculating the sensitivity of the desired cost function (e.g., ice volume).

Adjoint models, when used for these sensitivity calculations, are much more efficient because they can calculate a sensitivity "map" in a single step. Other than by analytical methods, adjoint models may be obtained from the underlying implementation as





GMDD
6, 5251–5288, 2013



a numerical program through a technique called algorithmic differentiation (AD). Heimbach and Bugnion (2009) used the AD tool TAF (Transformation of Algorithms in Fortran: Giering and Kaminski, 2003) to obtain an adjoint of the SICOPOLIS ice-sheet model (Greve, 1997, 2000). The adjoint was applied to the GrIS and used to calculate the sensitivity of the ice-sheet volume to basal sliding and basal melt rate.

In this paper, we present the adjoint of a thermomechanical ice-sheet model obtained using the OpenAD source-to-source transformation tool (Utke et al., 2006). OpenAD is developed at the University of Chicago and Argonne National Laboratory and is made available under an open source license. It can be downloaded from http://www.mcs.anl.gov/OpenAD/. Our motivation for this work is to study the sensitivity of the GrIS to changes in its boundary conditions, and we present some examples of results of output from this application.

The use of OpenAD in the present work has an important advantage compared with the previous study by Heimbach and Bugnion (2009). Their use of the commercial TAF AD tool meant that the adjoint model code could not be released to the community at large. In contrast, because OpenAD is freely available, we are able to provide the full source code of the forward model and its adjoint as a supplement to this paper. Our aim is to encourage glaciologists to use source-to-source translation by providing an accessible example that can be downloaded and compiled by any interested reader.

The paper is structured as follows. Section 2 presents the principles of AD. The forward thermomechanical model is described in Sect. 3. The adjoint of this model obtained with the OpenAD tool is then described in Sect. 4. Section 5 describes the verification of the forward model; its adjoint is tested against some selected numerical approximations in Sect. 6. Section 7 presents some examples of results obtained by applying the adjoint model to the GrIS.





GMDD
6, 5251–5288, 2013



## 2 Algorithmic differentiation

Calculation of model sensitivities is the central purpose of adjoint models. For a given numerical model (hereafter referred to as the *forward model*), sensitivities are the partial derivatives of a certain objective function – the cost function $C(\boldsymbol{x})$ – with respect to a set of control variables $\boldsymbol{u}$, where $\boldsymbol{x}$ are the state variables of the forward model. Thus, the cost function depends indirectly on the control variables. The derivatives $\partial C/\partial \boldsymbol{u}$ form the Jacobian of $C$. In practice, the cost function might be the volume of the ice sheet at a particular time or some other physical property of the system, and $\boldsymbol{u}$ might be the discrete values of a spatially varying parameter such as basal sliding parameter.

Sensitivities can be numerically approximated by perturbing each control variable $u_i$ in turn and calculating the corresponding elements of the Jacobian using, for example, the finite-difference method:

$$\frac{\partial C}{\partial u_i} \approx \frac{C(u_i + \Delta u_i) - C(u_i)}{\Delta u_i}. \tag{1}$$

The result becomes exact as $\Delta u_i$ approaches zero, but the effects of truncation error limit the precision with which sensitivities can be obtained by this method in finite precision arithmetic. Furthermore, the model must be run for each perturbation, thus making the approach computationally expensive (proportional to the number of control variables).

For nonlinear models the approximation error can be substantial given that one generally does not know the perturbation that minimizes the error in the trade-off between approximation and truncation errors. Obtaining derivatives that are exact would clearly be advantageous. This is what the adjoint model allows, because it is derived by differentiating the forward-model equations to obtain expressions for the Jacobian. However, for the large and complex numerical models often used in glaciology, manually deriving the adjoint would be time-consuming and prone to error.





GMDD
6, 5251–5288, 2013



Algorithmic differentiation is a technique that can be used to obtain sensitivities when the cost function $C$ and the model it relates to are specified as a computer program. AD tools take as input the source code for the forward model and add additional data (to hold the derivative values) and logic (to compute the derivative values) to the original program. AD may be implemented by source-to-source transformation or by operator overloading (see, e.g. Griewank et al., 1996). In this paper we apply the source transformation approach implemented in the OpenAD tool.

### 2.1 Theoretical basis of algorithmic differentiation

We do not describe the mathematical framework of adjoint models in depth here; our aim is to give enough background to support descriptions of specific technical aspects given later. A detailed discussion of AD is given, for example, by Griewank and Walther (2008). For an introduction in the context of the geosciences, the reader may refer to Wunsch and Heimbach (2007) and Heimbach and Bugnion (2009).

We assume that the forward model takes the form (Naumann and Utke, 2004)

$$\boldsymbol{y} = F(\boldsymbol{x}) \quad \text{with} \quad \boldsymbol{x} \in \Re^n, \boldsymbol{y} \in \Re^m, \tag{2}$$

where $\Re^n$ and $\Re^m$ are the spaces of input and output vectors $\boldsymbol{x}$ and $\boldsymbol{y}$, respectively, and $n$ and $m$ are the number of elements in each vector.

Scalar, first-order AD allows for computing projections of the Jacobian $J$ at a given point in the domain. In *forward mode* the projection is $\dot{\boldsymbol{y}} = J\dot{\boldsymbol{x}}$ for a given direction $\dot{\boldsymbol{x}}$, and in *reverse* or *adjoint mode* the projection is $\bar{\boldsymbol{x}} = J^T\bar{\boldsymbol{y}}$ for a given weight vector $\bar{\boldsymbol{y}}$ (cf. Lagrange multipliers). Note the connection to the usual adjoint operator definition with an inner product $\langle .,. \rangle$ when written as $\langle \bar{\boldsymbol{y}}, J\dot{\boldsymbol{x}} \rangle = \langle J^T\bar{\boldsymbol{y}}, \dot{\boldsymbol{x}} \rangle$.

In practical terms AD assumes $F$ is implemented as a program whose execution implies the execution of a sequence of elemental operations $\phi$ such as intrinsics (sin, cos, etc.) and arithmetic operators. For each elemental $r = \phi(u, v, \ldots)$ the propagation





GMDD
6, 5251–5288, 2013



of directional derivatives in forward mode means

$$\dot{r} = \frac{\partial \phi}{\partial u}\dot{u} + \frac{\partial \phi}{\partial v}\dot{v} + \dots \quad , \tag{3}$$

and similarly in reverse mode the propagation of adjoints implies

$$\begin{aligned} \bar{u} &= \bar{u} + \frac{\partial \phi}{\partial u}\bar{r} \\ \bar{v} &= \bar{v} + \frac{\partial \phi}{\partial v}\bar{r} \\ &\dots \end{aligned} \tag{4}$$

The adjoints of the arguments $u, v$ of $\phi$ are computed by using the adjoint of the return value $r$. Note that *forward mode* is also referred to as the *tangent linear model*.

From a modeling perspective, the cost function would not be considered part of the forward model; but for the purpose of computing sensitivities with AD, we may view them as one mathematical mapping with a scalar valued output. Therefore we have $F$ with the dimension of the output space $m = 1$, and we want to compute the gradient. Doing so in forward mode implies that one Jacobian projection yields one gradient element at a time and gives all gradient elements in one Jacobian projection. In other words, the forward-mode gradient has a computational cost factor of $O(n)$ over the original model computation, just like the finite differences in Eq. (1), but still yields derivatives with machine precision. In comparison, the computational cost factor for the reverse-mode gradient does not depend on $n$ at all and therefore is preferable.

The main caveat is the order in which the partials of $\phi$ are referenced. From Eq. (3) it is clear that one can simply compute them when $\phi$ is executed in the forward model. In contrast, when considering the computation order implied by Eq. (4), it becomes apparent that (as the name "reverse mode" suggests) the partials are referenced in the order opposite that of the computation of the $\phi$. Therefore one will have to devise a means to recover these values, for example by storing or recomputing them. The method used to achieve this is discussed in the next section.





GMDD
6, 5251–5288, 2013



## 2.2 Checkpointing

In theory one can store all the partial derivatives, or the arguments $u, v, \ldots$ to nonlinear $\phi$ needed to compute them, during the *storing* forward pass in preparation for the reverse pass. In practice the memory requirements for this *store-all* approach are frequently prohibitive. The other extreme, the recomputation from the initial state or *recompute-all* approach, implies an equally prohibitive computational overhead. A trade-off between storage and recomputation is possible by checkpointing (e.g., Griewank, 1992; Grimm et al., 1996). In the context of a time-stepping scheme with $t$ iterations as applicable in the ice-sheet model, we can rewrite Eq. (2) as the sequence of $t$ successive applications of $F$ to the the model state $\boldsymbol{x}$ starting with an initial state $\boldsymbol{x}_0$:

$$\boldsymbol{x}_t = F(F(\ldots F(\boldsymbol{x}_0)\ldots)), \quad \boldsymbol{x} \in \Re^n. \tag{5}$$

For simplicity we ignore the cost function $C$ and some initial setup. We may have enough main memory to store the values needed for partials for a sequence of $k \ll t$ time steps at a time. That means we have $s = \lfloor \frac{t}{k} \rfloor (+1)$ subsequences of length $k$ with a possible tail sequence (the "+1") if there is a nonzero remainder $t \bmod k$. Assuming the initial state $\boldsymbol{x}_0$ is available anyway, one can in a simple scheme write to disk $s-2$ checkpoints at iterations $ik, i = 1, \ldots, s-2$. Then, as one reaches iteration $k(s-1)$ one starts the storing forward pass for the last subsequence, immediately followed by the reverse pass. Then one restores the checkpoint written at iteration $k(s-2)$, followed by the storing forward pass from $k(s-2)$ to $k(s-1)$, immediately followed by the reverse pass for these time steps, and so on going backwards. In this way one would trade a recomputation effort of roughly one additional forward model execution for $(s-2)$ checkpoints on disk and main memory to store values for partials for $k$ instead of $t$ time steps at a time. For large $t$ the resulting $s$ may require too much checkpointing disk space. In these cases one can use a hierarchical scheme in which one first writes fewer checkpoints in distances $k' \gg k$ and then runs forward from the sparse checkpoints to write denser checkpoints until one reaches distance $k$.





GMDD
6, 5251–5288, 2013



## 2.3 Checkpointing with Revolve

Given a number $s$ of subsequences and the number $c$ of checkpoints that may be held (on disk) at any one time during the execution of the model and its adjoint, there is a binomial scheme that minimizes the recomputation overhead which is called the Revolve algorithm (Griewank and Walter, 2000).

The Revolve algorithm has been reimplemented in Fortran for use with the OpenAD tool. The recomputation overhead can be pre-computed as function of $s$ and $c$. For the computation in this paper we allowed $s \leq 20\,000$ checkpoints but as the checkpoint size depends on the size of the state vector $\boldsymbol{x}$, it, as well as $k$, has to be adapted to the computing environment and the dimension of the concrete problem being tackled.

Note that the computational overhead is an estimate that does not include the time for writing and reading the checkpoints under the assumption that those times are dominated by the time to compute the $k$ time steps of the respective subsequence. Using the Revolve implementation included with OpenAD requires the time-stepping loop in the original code to be changed to a loop in which the action in the loop body is governed by the Revolve algorithm.

## 2.4 The OpenAD tool

OpenAD is a source-to-source translation tool, which is made available under an open source license (Naumann and Utke, 2013). The tool is under continued development and maintenance at the University of Chicago and Argonne National Laboratory.

The code analysis built into the OpenAD tool provides, among other information, the sets of program variables that are referenced or modified within a given subroutine and its callees. This information is easily used to generate code that writes and reads checkpoints. Note that these checkpoints are typically significantly smaller than the process snapshots taken as "system-level" checkpoints and thus are more akin to "application-level" checkpoints without the need to manually create them. To use these automated checkpoints, one wraps the time-stepping loop body into a separate





GMDD
6, 5251–5288, 2013



subroutine. Similarly, if the time-stepping loop itself is also wrapped into a separate subroutine, then the OpenAD template mechanism can be used to replace the original time stepping loop with a loop that references the Revolve interfaces. These subroutine encapsulations have been done as permanent changes in the ADISM model code.

## 3 Forward ice-sheet model

This section describes the forward ice-sheet model (ISM). The model is a three-dimensional, finite-difference model, based on the established shallow ice approximation (SIA: Mahaffy, 1976; Jenssen, 1977). The two main components of the ISM solve, respectively, the evolution of ice thickness and temperature. The thickness calculation code was written from scratch but based on an existing model coded in Matlab (courtesy of T. Payne, personal communication, 2008), while the thermodynamic code is essentially the same as that contained in the Glimmer community ISM (Rutt et al., 2009), although minor changes were made for technical reasons. All the model code is written in Fortran 90.

### 3.1 Ice thickness calculation

The ice thickness equation is obtained by integrating vertically the continuity equation for an incompressible material (Van der Veen, 2002):

$$\frac{\partial H}{\partial t} = -\nabla \cdot \boldsymbol{q} + M, \tag{6}$$

where $\boldsymbol{q} = \bar{\boldsymbol{u}} H$ is the horizontal mass flux, $M$ is the surface mass balance rate, $H$ the ice thickness, $\bar{\boldsymbol{u}}$ the depth-mean horizontal velocity, and $\nabla$ is the horizontal gradient operator. Note that in the remainder of the paper, the symbols $n$, $s$, $\boldsymbol{u}$ and $x$ have different meanings from the earlier section covering the adjoint method.

The SIA assumes that surface and bedrock gradients are small, so the only important stress components are horizontal shear stresses. We can approximate the horizontal





GMDD
6, 5251–5288, 2013



shear stresses in $x$ and $y$ at level $z$ as (Rutt et al., 2009)

$$\tau_{xz} = -\rho g(s-z)\frac{\partial s}{\partial x}$$
$$\tau_{yz} = -\rho g(s-z)\frac{\partial s}{\partial y}, \quad (7)$$

where $\rho$ is the ice density, $g$ is the acceleration due to gravity, and $s$ is the surface elevation. The horizontal shear strain rate $\dot{\epsilon}_{xz}$ is

$$\dot{\epsilon}_{xz} = \frac{1}{2}\left(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x}\right), \quad (8)$$

and similarly for $\dot{\epsilon}_{yz}$. In practice, we assume that $\partial w/\partial x$ and $\partial w/\partial y$ are small compared with the vertical gradients of horizontal velocity, and we neglect them. These strain rates are related to stresses by the flow law. We use the standard Glen–Nye flow law (Glen, 1952; Nye, 1953), such that for horizontal shear stresses

$$\dot{\epsilon}_{xz} = A\tau_*^{n-1}\tau_{xz}, \quad (9)$$

where $A$ is a temperature-dependent coefficient, $n$ is a constant (usually taken to be 3), and the effective shear stress $\tau_*$ is given by the second invariant of the stress tensor for the SIA

$$\tau_* = (\tau_{xz}^2 + \tau_{yz}^2)^{\frac{1}{2}}. \quad (10)$$

The flow parameter $A$ depends on temperature according to the Arrhenius relationship

$$A(T^*) = fa \cdot \exp\left(\frac{-Q}{RT^*}\right), \quad (11)$$

where $R$ is the gas constant, $Q$ is the activation energy, $a$ is a temperature-independent constant of proportionality, and $f$ is a flow enhancement factor that can be used





GMDD
6, 5251–5288, 2013



to compensate for impurities and the development of anisotropic ice fabrics. The temperature $T^*$ is the absolute temperature $T$ corrected relative to the pressure melting point (Rutt et al., 2009):

$$T^* = T + \rho g C_t H, \tag{12}$$

where $C_t$ is the dependence of the melting point on pressure.

Combining the SIA stress balance Eq. (7) and the expression for strain rate Eq. (8) gives the vertical gradient of velocity:

$$\frac{\partial \boldsymbol{u}}{\partial z} = -2A(\rho g(s-z))^n |\nabla s|^{n-1} \nabla s. \tag{13}$$

Integrating Eq. (13) with respect to $z$ gives the vertical profile of $\boldsymbol{u}$:

$$\boldsymbol{u}(z) - \boldsymbol{u}(h) = -2(\rho g)^n |\nabla s|^{n-1} \nabla s \int_h^z A(s-z')^n \mathrm{d}z', \tag{14}$$

where $\boldsymbol{u}(h)$ is the basal velocity and $z'$ is a dummy vertical coordinate. A simple parameterization of basal sliding is included in the forward model, such that the basal velocity is proportional to the gravitational driving stress at the bed:

$$\boldsymbol{u}(h) = -B\rho g H \nabla s, \tag{15}$$

where $B$ is the sliding parameter. The value of $B$ can be made to depend on the presence of melt at the bed.

Integrating Eq. (14) again gives the averaged ice velocity $\bar{\boldsymbol{u}}$ (Rutt et al., 2009):

$$\bar{\boldsymbol{u}} = -\frac{2}{H}(\rho g)^n |\nabla s|^{n-1} \nabla s \int_h^s \int_h^z A(s-z')^n \mathrm{d}z' \, \mathrm{d}z - B\rho g H \nabla s. \tag{16}$$





GMDD
6, 5251–5288, 2013



It is convenient to express the continuity Eq. (6) in terms of diffusivity $D$,

$$\frac{\partial H}{\partial t} = -\nabla \cdot (D \nabla s) + M, \tag{17}$$

with diffusivity given by

$$D = 2(\rho g)^n |\nabla s|^{n-1} \int_h^s \int_h^z A (s - z)^n \mathrm{d}z' \, \mathrm{d}z + B \rho g H^2. \tag{18}$$

### 3.2 Numerical solution of thickness equation

The model is discretized on a rectangular domain whose nodes have a regular spacing in $x$ and $y$, such that $x_i = (i-1)\Delta x$ for $i \in [1, N_x]$, and similarly for $y_i$ with a node spacing of $\Delta y$.

The continuity equation written in terms of diffusivity Eq. (17) is solved to step the model forward in time. The expression for diffusivity Eq. (18) is discretized as follows:

$$D_{i,j} = 2(\rho g)^n A^*_{i,j} \left( \left[ \frac{s_{i+1,j} - s_{i-1,j}}{2\Delta x} \right]^2 + \left[ \frac{s_{i,j+1} - s_{i,j-1}}{2\Delta y} \right]^2 \right)^{\frac{n-1}{2}} H^{n+2}_{i,j} + B_{i,j} \rho g H^2_{i,j}, \tag{19}$$

where $A^*_{i,j}$ represents the effective value of the flow law coefficient $A$ for the ice column and is given by the vertical double integral in Eq. (18):

$$A^*_{i,j} = \int_h^s \int_h^z A_{i,j,k} (s - z')^n \mathrm{d}z' \, \mathrm{d}z, \tag{20}$$

where $k$ is the vertical coordinate and the integration is performed by using the trapezium rule.





GMDD
6, 5251–5288, 2013



The $x$ and $y$ components of the flux (denoted $q^x$ and $q^y$) are defined halfway between the main grid points, such that

$$q^x_{i+\frac{1}{2},j} = -\frac{(D_{i,j} + D_{i+1,j})}{2}\frac{(s_{i+1,j} - s_{i,j})}{\Delta x}, \tag{21}$$

$$q^y_{i,j+\frac{1}{2}} = -\frac{(D_{i,j} + D_{i,j+1})}{2}\frac{(s_{i,j+1} - s_{i,j})}{\Delta y}. \tag{22}$$

The continuity Eq. (17) is discretized by using an implicit time step:

$$\frac{H^{t+1}_{i,j} - H^t_{i,j}}{\Delta t} = \frac{q^{x,t+1}_{i+\frac{1}{2},j} - q^{x,t+1}_{i-\frac{1}{2},j}}{\Delta x} + \frac{q^{y,t+1}_{i,j+\frac{1}{2}} - q^{y,t+1}_{i,j-\frac{1}{2}}}{\Delta y} + M_{i,j}, \tag{23}$$

where $\Delta t$ is the time step and $t$ and $t+1$ indicate the time steps. In the case of fluxes at time $t+1$, the diffusivity is approximated as being equal to that at the current time step ($D^{t+1} \approx D^t$). Substituting in the expressions for the discretized flux (Eqs. 21, 22), setting $H^t_{i,j} = s^t_{i,j} + h^t_{i,j}$, and rearranging, we have

$$\alpha_{i,j}s^{t+1}_{i-1,j} + \beta_{i,j}s^{t+1}_{i+1,j} + \gamma_{i,j}s^{t+1}_{i,j-1} + \kappa_{i,j}s^{t+1}_{i,j+1} + \phi_{i,j}s^{t+1}_{i,j} = s^t_{i,j} + M_{i,j}\Delta t \tag{24}$$

with

$$\alpha_{i,j} = \frac{\left(D^t_{i+1,j} + D^t_{i,j}\right)}{2}\frac{\Delta t}{\Delta x^2}$$

$$\beta_{i,j} = \frac{\left(D^t_{i-1,j} + D^t_{i,j}\right)}{2}\frac{\Delta t}{\Delta x^2}$$

$$\gamma_{i,j} = \frac{\left(D^t_{i,j+1} + D^t_{i,j}\right)}{2}\frac{\Delta t}{\Delta y^2}$$





GMDD
6, 5251–5288, 2013



$$\kappa_{i,j} = \frac{\left(D^t_{i,j-1} + D^t_{i,j}\right)}{2} \frac{\Delta t}{\Delta y^2}$$

$$\phi_{i,j} = 1 - \alpha_{i,j} - \beta_{i,j} - \gamma_{i,j} - \kappa_{i,j}. \tag{25}$$

Equation (24) can be thought of as a matrix equation of the form

$$\mathbf{A}\boldsymbol{s}^{t+1} = b, \tag{26}$$

where $\mathbf{A}$ is a sparse matrix composed of the elements $\alpha_{i,j}$, $\beta_{i,j}$, $\gamma_{i,j}$, $\kappa_{i,j}$ and diagonal elements $\phi_{i,j}$, and $b$ is the right-hand side of Eq. (24).

Equation (26) is solved by using the UMFPACK solver (Davis, 2011). This solver computes the LU factors in the factorization stage, and Eq. (26) is then solved without iterative refinement.

### 3.3 Temperature evolution equation

The evolution of temperature in an ice mass is governed by three processes: thermal diffusion, advection, and strain heating. In common with many three-dimensional ice-sheet models, we define a new vertical coordinate ($\sigma$), such that $\sigma = 0$ at the ice surface and $\sigma = 1$ at the bed:

$$\sigma = \frac{s - z}{H} \quad . \tag{27}$$

For a complete description of the resulting coordinate transformations, the reader is referred to Pattyn (2003).

Expressed in sigma coordinates, the temperature equation is as follows:

$$\frac{\partial T}{\partial t} = \frac{k}{\rho c_p H^2}\frac{\partial^2 T}{\partial \sigma^2} - \mathbf{u}\cdot\nabla T + \frac{\sigma g}{c_p}\frac{\partial \boldsymbol{u}}{\partial \sigma}\cdot\nabla s + \frac{1}{H}\frac{\partial T}{\partial \sigma}(w - w_{\text{grid}}), \tag{28}$$





GMDD
6, 5251–5288, 2013



where $T$ is the absolute temperature, $k$ is the thermal conductivity of ice, and $c_p$ is the specific heat capacity of ice. The coordinate transformation introduces a number of terms due to the time-variation of the ice geometry: these can be thought of as a vertical grid velocity and are combined into the term $w_{\text{grid}}$ (Rutt et al., 2009).

Because the vertical gradients of temperature in an ice sheet tend to be significantly larger than the horizontal gradients, only vertical diffusion of heat is represented in the model. Both horizontal and vertical advection of heat are represented, however. The third term in Eq. (28) represents the strain heating effect. This is of key importance to the thermomechanical effects seen in ice sheets.

### 3.4 Numerical solution of temperature equation

The code of the temperature evolution equation comes from the Glimmer ice-sheet model (Rutt et al., 2009). Details of the discretization of each of the terms in Eq. (28) are given by Hagdorn et al. (2010).

The temperature is solved iteratively for each column of ice; temperatures not in the column are taken from the previous iteration. The remaining unknown temperatures (each of the terms on the left of Eq. 28) are discretized by using the Crank–Nicolson scheme to form a tridiagonal matrix equation. The temperature is set to never exceed the pressure melting point.

## 4 Adjoint model

The adjoint model of the forward thermomechanical model described in Sect. 3 was obtained with the OpenAD tool (Utke et al., 2006). In order for the code to be passed through the tool, some changes need to be made to the model code and the build system.





GMDD
6, 5251–5288, 2013



### 4.1 Preparation for use of OpenAD

The user has to supply information regarding the inputs and outputs, adapt the *driver* logic to make use of the derivatives and modify the build system to execute the transformation itself and compile/link the transformed source code.

#### 4.1.1 Pragmas

The way in which OpenAD treats certain portions of the model source code is controlled by pragmas inserted into the code. Specifically, the treatment of a given subroutine in the context of the checkpointing scheme is determined by applying templates, that are specified by pragmas.

Likewise, in the source code, one has to identify which program variables correspond to the inputs and outputs of the model (as in Eq. 2). This step is accomplished by using pragmas to declare given program variables to be the *independent* and *dependent* variables of the model (the control parameters and cost function, respectively). The choice of independent and dependent variables depends on the application of the adjoint model; details are explained by Utke et al. (2013).

#### 4.1.2 Makefiles and drivers

In common with most numerical models, ADISM has a top-level driver routine that governs model initialization, execution, and result retrieval. The model code is compiled and linked by using *GNU Make*, which coordinates the build procedure. For the computation of the derivatives, the driver routine has to be adjusted to trigger the start of computation and the retrieval of the computed sensitivities. Because a source transformation step is involved, the build procedure has to be expanded to perform the transformation steps and then build the model from the transformed sources. The transformation takes places in multiple stages related to the different tool components involved (Fig. 1b).





GMDD
6, 5251–5288, 2013



Where OpenAD is being used for the transformation of a simple model, the transformation stages may be wrapped into a script. For nontrivial use, as in the case of ADISM, the transformation stages benefit from being kept explicit because the transformation can be modified at each of the stages. Examples are given by Utke et al. (2013).

The principal flow of operation is as follows:

- The code of a model $f$ is parsed by the *Open64* front-end. This produces a binary file, which contains the Open64-internal representation in the so-called *whirl* format ($f_{\text{whirl}}$).
- This representation is analyzed by the *OpenAnalysis* component, and results are translated into a language-independent XML format called *xaif* ($f_{\text{xaif}}$).
- This language-independent representation is then transformed ($f'_{\text{xaif}}$) by *Xaifbooster*, which is the core AD component, into either the adjoint or the tangent linear model.
- The *whirl* representation of this differentiated function, $f'_{\text{whirl}}$, is obtained.
- The Open64 front-end unparses into Fortran code the *whirl* representation to the differentiated function $f'$.

In the case of ADISM, the driver routine was adapted to provide variants appropriate to both the adjoint and tangent linear models. We provide the code of both the original model and the adjoint in the Supplement to this paper, as well as the makefiles necessary to run OpenAD.

### 4.2 Changes to the source code

The OpenAD framework, its components, and the Fortran language standard itself are under continued development. Language features are covered based on user demand and weighing their implementation effort in comparison with the effort for workarounds





GMDD
6, 5251–5288, 2013



in the user code. Ancillary logic or calls to third party libraries may require the use of stubs. Implementing workarounds and stubs results in certain changes to the model source code.

#### 4.2.1 Language limitations

The OpenAD version used in this paper has a number of limitations with respect to the subset of Fortran 90 syntax that can be handled. These limitations are due to the use of the Open64/SL front-end; consequently, current OpenAD development work is focused on replacing Open64/SL with the Rose front-end (ROSE compiler infrastructure, 2013), with which many language limitations are eliminated.

One aspect of Fortran 90 syntax not currently supported by OpenAD is the *derived type*. Derived types were used extensively in the Glimmer subroutines that form the thermal part of ADISM (Rutt et al., 2009). The Glimmer code was changed to eliminate derived types; instead, extra parameters were passed to subroutine calls.

A second unsupported Fortran 90 language construct is the *where* construct. Likewise, these were rewritten by using *if* blocks and loops.

In addition, the Fortran *implied do* statement, usually used to construct arrays in a compact way, was replaced by explicit *do* loops wherever it occurred.

#### 4.2.2 Stubs

Not all the code that constitutes a numerical model is part of the computation to be differentiated. In our case this situation applies to the I/O logic implemented by the NetCDF library where the source code may not be in Fortran, may not be available, or may use language features not well supported by the transformation tool and thereby may unnecessarily complicate the transformation process. In such cases it is appropriate to have the subroutine in question (here from the NetCDF library) be represented by a *stub*, that is, a subroutine with the same interface and a body that represents only the simplified differentiable data dependencies of the parameters,





GMDD
6, 5251–5288, 2013



if any. Prior to the final compilation of the transformed code, the stub routines are removed. If the stubs do not represent data dependencies, one need simply link to the original implementations in the NetCDF libraries.

Stubs are also used to implement adjoint propagation of high-level mathematical operations such as the linear solution of the ice thickness equation using UMFPACK (Davis, 2011). Here the subroutine parameters and therefore the stub have differentiable dependencies. Rather than trying to differentiate by brute force through the UMFPACK library and also recognizing that the recent versions of UMFPACK are implemented in C rather than Fortran, we inject a stub for the solve step. Prior to the final compilation of the transformed code we apply the OpenAD template mechanism to replace that stub with the logic needed to carry out the appropriate action during the forward sweep (the system solve) and the adjoint sweep (the system solve with the transpose). Not only does this approach permit the use of UMFPACK as a black-box library written in C, but it is also more efficient than the brute-force differentiation.

### 4.3 Memory optimization

In order to optimize memory use and computational cost, the Revolve checkpointing scheme (Sect. 2.3) was used in the model. The code of the forward model was structured so that the main time loop was in a separate file. Checkpointing was used for subroutines in the time loop following the Revolve scheme. No checkpointing was used for those subroutines outside the main time loop, a technique also known as split-mode AD.

### 4.4 Build environment

The model runs in this paper were undertaken using an 64-bit Ubuntu-derived Linux distribution, using the GCC (`gfortran`) compiler version 4.7.3.





GMDD
6, 5251–5288, 2013



## 5 Forward model verification

In this paper, we use the term *verification* to refer to the process of checking that the forward model code has been implemented correctly. We do not address the question of *validation*, which concerns the extent to which the underlying mathematical model is a good analogue for the target system. In any case, the strengths and weaknesses of finite-difference SIA models are well known and are covered at length elsewhere (e.g., van den Berg et al., 2006; Blatter et al., 2011)

ADISM is first verified against the well-known EISMINT2 benchmark (Payne et al., 2000). The model applied to the GrIS is then verified against the EISMINT3 benchmark (Huybrechts, 1997, unpublished). In both cases, the verification is made by comparison with the performance of the numerical models included in the original studies.

### 5.1 Comparison with EISMINT2 benchmark

A domain of 1500 km × 1500 km and a horizontal resolution of 25 km are used in the EISMINT2 benchmark. ADISM was configured according to Experiments A and H (Payne et al., 2000). In both experiments, a circular ice sheet is grown on a flat bed under idealized forcing. Experiment A is the most basic configuration, where basal sliding is disabled; in Experiment H, basal sliding is enabled for regions where the bed is at pressure melting point.

In both experiments, the surface mass balance $M$ has a maximum value $M_{\max}$ in a circular region at the center of the domain, but beyond that it decreases with distance from the center. It is expressed as

$$M(x,y) = \min\left(M_{\max}\,,\; S_{\mathrm{b}}\left[R_{\mathrm{e}} - \sqrt{(x-x_{\mathrm{s}})^2 + (y-y_{\mathrm{s}})^2}\;\right]\right), \tag{29}$$

where $(x_{\mathrm{s}}, y_{\mathrm{s}})$ is the location of the center of the domain, $R_{\mathrm{e}} = 450$ km is the radial distance from the center at which the surface mass balance becomes zero, and $S_{\mathrm{b}} = 0.01\,\mathrm{m\,yr^{-1}\,km^{-1}}$ is the gradient of change of surface mass balance with horizontal





GMDD
6, 5251–5288, 2013



distance. The surface temperature $T_{surf}$ is expressed as

$$T_{surf} = T_{min} + S_t\sqrt{(x - x_s)^2 + (y - y_s)^2}, \quad (30)$$

where $T_{min} = 238.15\,K$ is the minimum surface air temperature and $S_t = 1.67 \times 10^{-2}\,K\,km^{-1}$ is the gradient of air-temperature change with horizontal distance.

In Experiment H, the linear sliding law given in Eq. (15) was used, with $B = 10^{-3}\,m\,Pa^{-1}\,yr^{-1}$. Basal sliding was enabled only where the basal ice is at pressure melting point.

For both experiments, the model was run forward from zero ice initial conditions for 200 ka. The verification was carried out by comparing values of ice divide thickness and basal temperature at the end of the run with those of the models that took part in the EISMINT2 benchmark (Payne et al., 2000) (Table 1). The range and mean are shown for the EISMINT2 models. Values of basal temperatures of the ADISM for both experiments (A and H) are within those of the EISMINT2 range. The value of ice divide thickness of Experiment A is 40 m lower (about 1.2 % lower than the actual value) than the EISMINT 2 range, while that of Experiment H is within this range.

From a physical point of view, Experiment H is more realistic than Experiment A, because it includes basal sliding where ice is temparate. Basal temperatures and ice divide thickness of Experiment H both being within the EISMINT2 range (Table 1) gives us confidence that the model is implemented correctly for the physically realistic case, which is desired for the application of the forward model to Greenland (Sect. 5.2).

We note that the variation between models in Payne et al. (2000) is larger for Experiment H than it is for Experiment A. The reason is that the thermomechanical feedback between ice flow, temperature, and sliding causes instabilities. These instabilities manifest themselves as unsteady radial fast flow features embedded in the ice sheet (Payne and Baldwin, 2000), as are seen in the field of horizontal velocity when ADISM is configured according to Experiment H (Fig. 2a). Instabilities are also seen as spokes of warm and cold ice in the basal temperature field (Fig. 2b).





GMDD
6, 5251–5288, 2013



### 5.2 Comparison with EISMINT3 Greenland benchmark

As an example of an application to a real ice-sheet geometry, ADISM was configured for the GrIS. Again, it was necessary to verify that the model performed as expected for this geometry, and so a comparison was made with the GrIS configuration from the unpublished EISMINT3 benchmark (Huybrechts, 1997).

The EISMINT3 Greenland benchmark uses a spatial resolution of 20 km in $x$ and $y$ and a time step of 10 yr. The initial ice geometry and bed DEM are those of Letreguilly et al. (1991). The surface mass balance is calculated by using an annual positive degree day (PDD) model (Reeh, 1991); this requires inputs of annual precipitation, mean annual surface temperature ($T_a$), and mean summer temperature ($T_s$). The annual precipitation field is that compiled by Ohmura and Reeh (1991). The temperature forcing is specified as a pair of simplified parameterizations (Huybrechts and deWolde, 1999; Ritz et al., 1997), based on observational data (Ohmura, 1987):

$$T_a = 49.13 - 0.007992\,(\max(s, 20(\Phi - 65)) - 0.7576\,\Phi, \quad (31)$$

$$T_s = 30.38 - 0.006277\,s - 0.3262\,\Phi, \quad (32)$$

where $s$ is the ice surface elevation (m) and $\Phi$ is latitude (° N). We are aware that these data are neither the most recent nor the highest quality available and that the dynamic characteristics of the modeled ice sheet are sensitive to the choice of ice geometry and forcing climate (Stone et al., 2010). Nevertheless, we adopt them for the present work because they allow comparison with the EISMINT3 results.

The EISMINT3 benchmark uses a flow enhancement factor of $f = 3$ in Eq. (11). Calving is taken into account by assuming ice is removed when the thickness is within 200 m of the floatation point. The geothermal heat flux is 0.05 W m$^{-2}$, and basal sliding is disabled.

The model was run forward from the measured geometry for 100 ka, and the surface compared with those modeled as part of the EISMINT3 exercise. Unfortunately, no original data were available for these model runs, and so comparison had to be





GMDD
6, 5251–5288, 2013



undertaken using the plots published in Huybrechts (1997): in this respect, ADISM was seen to perform satisfactorily. The maximum surface elevation was within the range of those obtained in the EISMINT3 benchmark. The surface elevation and volume with and without basal sliding also compared well with present-day values (Table 2). From the verification tests in this section, we can have confidence in the ADISM as applied to the GrIS.

## 6 Testing the adjoint model

To test the adjoint model, we check the adjoint sensitivities against those calculated by using finite differences from the forward model (as in Eq. 1). Forward model sensitivities should not be confused with those of the *forward mode* (or tangent linear mode) of AD. In contrast to the verification of the forward model, basal sliding is enabled in these model runs. A ten year time step was used in the calculation of sensitivities.

We compare sensitivities of both the EISMINT2 and EISMINT3 cases. In both cases, we run the forward model to a quasi-steady state before calculating sensitivities over a further 100 yr of model runs. The reason is that the model exhibits highly nonlinear behavior when run from an artificial initial state. Using an initial "spin-up" period allows the model to reach a more physically consistent state before sensitivities are calculated. In each case, basal sliding is included in both spin up and model runs.

Calculating approximate sensitivities by using the forward model is difficult because even small changes in the magnitude of the perturbation yield can different orders of magnitude and even sign changes of the derivative approximations. Using an adjoint model, one can compute machine-precision sensitivities with respect to that artificial initial state. However, the large gradients of the initial nonlinearities render the result useless for practical purposes. It is therefore crucial to conduct the adjoint experiments from a spun-up model state.





GMDD
6, 5251–5288, 2013



### 6.1 Adjoint of the model configured according to the EISMINT2 benchmark

We compared sensitivities with the model configured according to Experiment A and H of the EISMINT2 benchmark. While the models in the EISMINT2 benchmark are run over 200 kyr, the thermal regime has reached steady state after about 150 kyr. Sensitivities were run from the state of the forward model after 150 kyr, which is run from zero ice thickness.

Presented here is the comparison of sensitivities of the model configured according to the more realistic setup of Experiment H. The comparison of those of the model of Experiment A is also referred to for completeness. The adjoint and forward sensitivity maps of ice-sheet volume to surface mass balance for Experiment H are shown in Fig. 3a and b. Here, surface mass balance is a two-dimensional independent variable (covering the model domain). Each point on the sensitivity maps is the total change of volume at the end of the model run (100 yr) for a unit perturbation of surface mass balance at that point.

The difference between the adjoint and forward sensitivities (Fig. 3c) is negligible in the center ($10^{-9}\times$ the actual value) and is about 0.2 % the actual value at the margins. This is good, considering the large gradient of sensitivity at the margins. This small difference is a reflection of the fact that the model as considered here is very stable, with relatively little nonlinear behavior, which would otherwise give rise to larger differences between the forward and adjoint sensitivities.

For Experiment A, the adjoint model also performed well, with a negligible difference at the center and 0.05 % near the margins. From these tests we can have confidence in the adjoint of the thermomechanical model, so that the adjoint of the thermomechanical model applied to the GrIS can be tested.





GMDD
6, 5251–5288, 2013



### 6.2 Adjoint of the model applied to the Greenland Ice Sheet

Sensitivities are calculated starting from the state of the forward model applied to the GrIS after 100 kyr. Sliding is taken into account in the spin up run – this contrasts with the EISMINT3 case which does not include basal sliding.

When examining sensitivities, it is desirable that the ice sheet from which the model run starts is as close to the present day ice sheet as possible. Using a flow enhancement factor of 2 resulted in the optimal ice sheet; sensitivities were run from this state (Fig. 5b). The ice sheet could be tuned further to the present day ice sheet using data assimilation; this is beyond the scope of this paper, however.

The adjoint sensitivity of volume to surface mass balance (Fig. 4a) compares very well with respect to the forward model sensitivity (Fig. 4b). The difference between these is about 0.01 % at the center and up to 0.05 % in the northeastern region (Fig. 4c). The larger difference is evidence of the nonlinearity in the model. In this case a perturbation factor of $10^{-8}$ was used.

We note here that in the forward sensitivity approximations, a uniform perturbation is often applied over the whole domain, causing variations in the approximation-to-truncation trade-off. Even if the optimal trade-off could be found for the perturbation separately at every discretization point in the domain, the forward approximation would still lose half the significant digits. For reference, the initial spin-up state from which the spun-up sensitivities are run is shown in Fig. 5b, along with the observed surface elevation in Fig. 5a.

Because the adjoint sensitivities agree well with the approximate sensitivities calculated from the forward model, these validation tests give a high degree of confidence in the adjoint model as applied to the GrIS.





GMDD
6, 5251–5288, 2013



## 7 Discussion

Motivation for this work arose from the computational efficiency of the adjoint model and the power of AD for obtaining sensitivities. The reality of this is demonstrated by the fact that the computational time of the GrIS adjoint sensitivities (Fig. 4a) was about 500 times less than that required to compute the sensitivities by using the forward model.

The adjoint model is also useful in that it gives exact sensitivities, which have no numerical error. Differences between the forward and adjoint sensitivities in Sect. 6 are due to the choice of perturbation factor used in the forward sensitivities, the finite-difference approximation of forward sensitivities, and not the adjoint sensitivities.

The purpose of this paper was the presentation and testing of an adjoint model. This work has enabled further study of adjoint ice-sheet sensitivities, with the development of an adjoint model using a freely accessible open source AD tool.

## 8 Conclusions

This paper presented a forward ice-sheet model applied to the GrIS and its adjoint model obtained by using the OpenAD tool. The adjoint model was also tested against the forward model. The forward model was verified and performed well against the EISMINT3 benchmark. Certain changes to the code of the forward model needed to be carried out to make it compatible with the OpenAD tool, so that an adjoint model could be obtained from it. The adjoint model performed well with respect to the forward model.

**Supplementary material related to this article is available online at http://www.geosci-model-dev-discuss.net/6/5251/2013/gmdd-6-5251-2013-supplement.zip.**





GMDD
6, 5251–5288, 2013



*Acknowledgements.* This work was initiated during a research visit to MIT and was pursued further during a visit to Argonne National Laboratory, funded by the Leverhulme Trust GLIMPSE project. J. McGovern thanks Patrick Heimbach for initiating the work with OpenAD. This work was carried out during J. McGovern's Ph.D. research, also funded by the Leverhulme Trust. We thank Tony Payne for supplying the original Matlab code for the thickness solver. Utke was supported by the US Department of Energy, under contract DE-AC02-06CH11357.

The publication of this article is financed by CNRS-INSU.

## References

Blatter, H., Greve, R., and Abe-Ouchi, A.: Present state and prospects of ice sheet and glacier modelling, Surv. Geophys., 32, 555-583, doi:10.1007/s10712-011-9128-0, 2011. 5271

Brinkerhoff, D., Meierbachtol, T., Johnson, J., and Harper, J.: Sensitivity of the frozen/melted basal boundary to perturbations of basal traction and geothermal heat flux: Isunnguata Sermia, western Greenland, Ann. Glaciol., 52, 43–50, 2011. 5253

Cazenave, A.: How fast are the ice sheets melting?, Science, 314, 1250–1252, 2006. 5253

Courtier, P., Derber, J., Errico, R., Louis, J.-F., and Vukićević, T.: Important literature on the use of adjoint, variational methods and the Kalman filter in meteorology, Tellus A, 45, 342–357, doi:10.1034/j.1600-0870.1993.t01-4-00002.x, 1993. 5253

Davis, T.: UMFPACK: unsymmetric multifrontal sparse LU factorization package, available at: http://www.cise.ufl.edu/research/sparse/umfpack/, last access: July 2010, 2011. 5265, 5270

Giering, R. and Kaminski, T.: Applying TAF to generate efficient derivative code of Fortan 77-95 programs, Proceedings in Applied Mathematics and Mechanics (PAMM), 2, 54–57, 2003. 5254

Glen, J.: Experiments on the deformation of ice, J. Glaciol., 2, 111–114, 1952. 5261





GMDD

6, 5251–5288, 2013





Goldberg, D. N. and Sergienko, O. V.: Data assimilation using a hybrid ice flow model, The Cryosphere, 5, 315–327, doi:10.5194/tc-5-315-2011, 2011. 5253
Greve, R.: Application of a polythermal three-dimensional ice sheet model to the Greenland ice sheet: response to steady-state and transient climate scenarios, J. Climate, 10, 901–918, 1997. 5254
Greve, R.: On the response of the Greenland ice sheet to climate change, Climatic Change, 46, 289–303, 2000. 5254
Griewank, A.: Achieving logarithmic growth of temporal and spatial complexity in reverse automatic differentiation, Optim. Method. Softw., 1, 35–54, 1992. 5258
Griewank, A. and Walter, A.: Algorithm 799: Revolve: an implementation of checkpointing for the reverse or adjoint mode of computational differentiation, ACM T. Math. Software (TOMS), 26, 1–27, 2000. 5259
Griewank, A. and Walther, A.: Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation, no. 105, in: Other Titles in Applied Mathematics, 2nd Edn., SIAM, Philadelphia, PA, available at: http://www.ec-securehost.com/SIAM/OT105.html, last access: December 2010, 2008. 5256
Griewank, A., Juedes, D., and Utke, J.: Algorithm 755: ADOL-C: A package for the automatic differentiation of algorithms written in C/C++, ACM T. Math. Software (TOMS, 22, 131–167, doi:10.1145/229473.229474, 1996. 5256
Grimm, J., Pottier, L., and Rostaing-Schmidt, N.: Optimal time and minimum space-time product for reversing a certain class of Programs, in: Computational Differentiation: Techniques, Applications, and Tools, edited by: Berz, M., Bischof, C. H., Corliss, G. F., and Griewank, A., SIAM, Philadelphia, PA, 95–106, 1996. 5258
Hagdorn, M., Rutt, I., Payne, T., and Hebeler, F.: Glimmer 1.5.1 Documentation, Tech. rep., 89 pp., Edingburgh University, UK, 2010. 5266
Heimbach, P. and Bugnion, V.: Greenland ice sheet volume sensitivity to basal, surface, and initial conditions, derived from an adjoint model, Ann. Glaciol., 50, 67–80, 2009. 5254, 5256
Huybrechts, P.: Report on the third EISMINT workshop on model intrercomparision, Grinderwald, Switzerland, 25–27 September 1997. 5271, 5273, 5274
Huybrechts, P. and deWolde, J.: The dynamic response of the Greenland ice sheet to multiple-century climatic warming, J. Climate, 12, 2169–2188, 1999. 5273
Jenssen, D.: A three-dimensional polar ice-sheet model, J. Glaciol., 18, 373–389, 1977. 5260





GMDD
6, 5251–5288, 2013



Lahoz, W., Khattatov, B., and Menard, R. (Eds.): Data Assimilation, Springer, Berlin, Heidelberg, available at: http://www.springerlink.com/content/978-3-540-74702-4/contents/, last access: December 2012, 2010. 5252
Letreguilly, A., Huybrechts, P., and Reeh, P.: Steady-state characterisitics of the Greenland ice sheet under different climates, J. Glaciol., 37, 149–157, 1991. 5273
MacAyeal, D.: The basal stress distribution of ice stream E, Antarctica, inferred by control methods, J. Geophys. Res., 97, 595–603, 1992. 5253
Mahaffy, M. W.: A three-dimensional numerical model of ice sheets: tests on the Barnes Ice Cap, Northwest territories, J. Geophys. Res., 81, 1059–1066, doi:10.1029/JC081i006p01059, 1976. 5260
Naumann, U. and Utke, J.: Source Templates for the Automatic Generation of Adjoint Code Through Static Call Graph Reversal Computational Science – ICCS 2005, Lect. Notes Comput. Sc., 3514, 338–346, 2005. 5256
Naumann, U. and Utke, J.: OpenAD, available at: http://www.mcs.anl.gov/OpenAD (last access: 28 January 2013), 2013. 5259
Nye, J. F.: The flow law of ice from measurements in glacier tunnels, laboratory experiments and the Jungfraufirn Borehole Experiment, P. Roy. Soc. Lond. A Mat., 219, 477–489, doi:10.1098/rspa.1953.0161, 1953. 5261
Ohmura, A.: New temperature distribution maps for Greenland, Z. Gletscherkd. Glazialgeol., 23, 1–45, 1987. 5273
Ohmura, A. and Reeh, N.: New precipitation and accumulation maps for Greenland, J. Glaciol., 37, 140–148, 1991. 5273
Pattyn, F.: A new 3-dimensional higher-order thermomechanical ice sheet model: basic sensitivity, ice stream development, and ice flow across subglacial lakes, J. Geophys. Res., 108, 1–15, doi:10.1029/2002JB002329, 2003. 5265
Payne, A. J. and Baldwin, D. J.: Analysis of ice-flow instabilities identified in the EISMINT intercomparison exercise, Ann. Glaciol., 30, 204–210, doi:10.3189/172756400781820534, 2000. 5272
Payne, A. J., Huybrechts, P., Abe-Ouchi, A., Calov, R., Fastcook, R., G., Marshall, S., Marsiat, I., Ritz, C., Tarasov, L., and Thomassen, M. P. A.: Results from the EISMINT model intercomparision: the effects of thermodynamic coupling, J. Glaciol., 46, 227–238, 2000. 5271, 5272





GMDD
6, 5251–5288, 2013



Reeh, N.: Parameterization of melt rate and surface temperature on the Greenland ice sheet, Polarforschung, 59, 113–128, 1991. 5273

Ritz, C., Fabre, A., and Letreguilly, A.: Sensitivity of a Greenland model to ice flow and ablation parameters: consequences for the evolution through the last glacial cycle, Clim. Dynam., 13, 11–23, 1997. 5273

ROSE compiler infrastructure: available at: http://www.rosecompiler.org/ (last access: March 2013), 2013. 5269

Rutt, I. C., Hagdorn, M., Hulton, N. R. J., and Payne, A. J.: The "Glimmer" community ice sheet model, J. Geophys. Res., 114, F02004, doi:10.1029/2008JF001015, 2009. 5260, 5261, 5262, 5266, 5269

Stone, E. J., Lunt, D. J., Rutt, I. C., and Hanna, E.: Investigating the sensitivity of numerical model simulations of the modern state of the Greenland ice-sheet and its future response to climate change, The Cryosphere, 4, 397–417, doi:10.5194/tc-4-397-2010, 2010. 5273

Utke, J., Naumann, U., Fagan, M., Tallet, N., Strout, M., Heimbach, P., Hill, C., and Wunsch, C.: OpenAD/F: a modular, open-source tool for automatic differentiation of Fortran, ACM T. Math. Software (TOMS), 34, 1–34, 2006. 5254, 5266, 5284

Utke, J., Naumann, U., and Lyons, A.: OpenAD/F: User Manual, Tech. rep., Argonne National Laboratory, latest version available at: http://www.mcs.anl.gov/OpenAD/openad.pdf, last access: July 2012, 2013. 5267, 5268

van den Berg, J., van de Wal, R., and Oerlemans, J.: Effects of spatial discretization in ice-sheet modelling using the shallow-ice approximation, J. Glaciol., 52, 89–98, doi:10.3189/172756506781828935, 2006. 5271

Van der Veen, C.: Polar ice sheets and global sea level: how well can we predict the future?, Global Planet. Change, 32, 165–194, 2002. 5260

Wunsch, C. and Heimbach, P.: Practical global oceanic state estimation, Physica D, 230, 197–208, doi:10.1016/j.physd.2006.09.040, 2007. 5256





GMDD

6, 5251–5288, 2013



**Table 1.** Steady-state ice divide thickness (m) and basal temperatures (°C) of the forward thermomechanical model (ADISM) and of models that took part in Experiments A and H of the EISMINT2 benchmark. Mean and range of values of EISMINT2 are shown.

| | | Experiment A | Experiment H |
|---|---|---|---|
| Thickness (m) | ADISM | 3603.23 | 3475.62 |
| | EISMINT2 mean | 3688.342 | 3507.984 |
| | EISMINT2 range | 3644.000–3694.450 | 3433.100–3645.320 |
| Basal temp. (°C) | ADISM | −17.31 | −17.72 |
| | EISMINT2 mean | −17.55 | −17.95 |
| | EISMINT2 range | −18.68 to −16.06 | −21.45 to −17.05 |





GMDD
6, 5251–5288, 2013



**Table 2.** GrIS steady-state maximum surface elevation (m) and volume ($10^{15}$ m$^3$), with and without basal sliding, and present day values.

| | Sliding | No Sliding | Present Day |
|---|---|---|---|
| Max. surface elevation (m) | 3143 | 3251 | 3216 |
| Volume ($10^{15}$ m$^3$) | 3.04 | 3.16 | 2.86 |





GMDD
6, 5251–5288, 2013



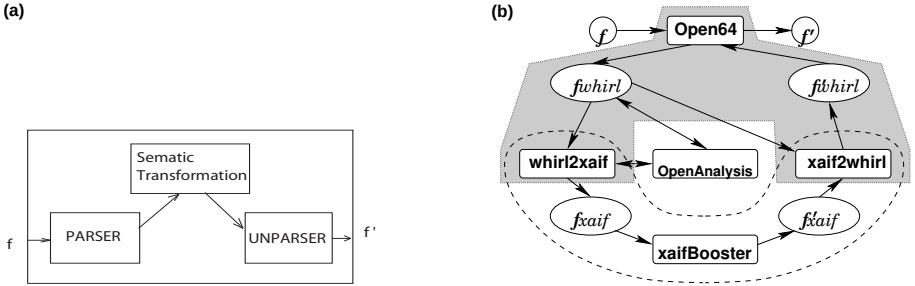


**Fig. 1. (a)** Source-to-source AD tool. The code of model is parsed, transformed, and unparsed. **(b)** Components in the OpenAD tool (Utke et al., 2006).





GMDD
6, 5251–5288, 2013



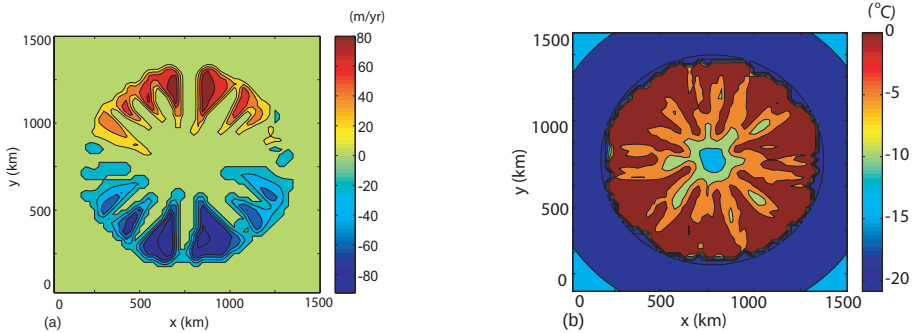


**Fig. 2.** Effects of thermomechanical coupling seen in **(a)** horizontal $u$ velocity (m yr$^{-1}$), **(b)** basal temperature (°C) fields of EISMINT2 benchmark, Experiment H ($\Delta x$: 25 km; $\Delta t$: 10 yr).





GMDD
6, 5251–5288, 2013



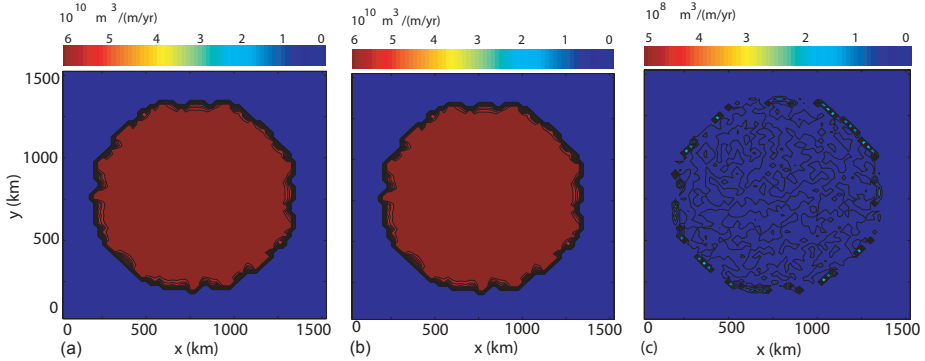


**Fig. 3. (a)** Adjoint and **(b)** forward sensitivity map of ice-sheet volume to surface mass balance $(m^3\,(m\,yr^{-1})^{-1})^{-1}$ of the thermomechanical ice-sheet model, and **(c)** difference between adjoint and forward sensitivities. The model configuration is EISMINT2, Experiment H. Contour interval in **(a)** and **(b)** $2 \times 10^9\,m^3\,(m\,yr^{-1})^{-1}$, in **(c)** $2 \times 10^7\,m^3\,(m\,yr^{-1})^{-1}$ ($\Delta x$: 25 km; $\Delta t$: 10 yr).





GMDD
6, 5251–5288, 2013



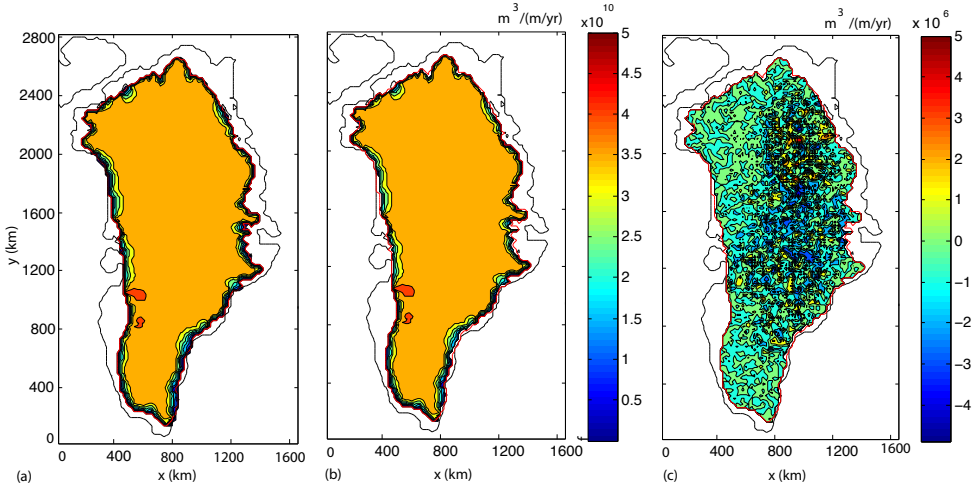


**Fig. 4. (a)** Adjoint and **(b)** forward sensitivity maps of volume to surface mass balance $(m^3\,(m\,yr^{-1})^{-1})^{-1}$ over 100 yr of the thermomechanical model applied to the Greenland ice sheet. **(c)** Difference between adjoint and forward volume sensitivity to surface mass balance $(m^3\,(m\,yr^{-1})^{-1})^{-1}$. Red line is the edge of the ice sheet, black line is the coast. Contour interval: **(a)** and **(b)** $5 \times 10^9\,m^3\,(m\,yr^{-1})^{-1})^{-1}$; **(c)** $10^6 m^3\,(m\,yr^{-1})^{-1}$ ($\Delta x$ : 20 km; $\Delta t$: 10 yr).





GMDD
6, 5251–5288, 2013

ADISM: ice sheet adjoint model

J. McGovern et al.



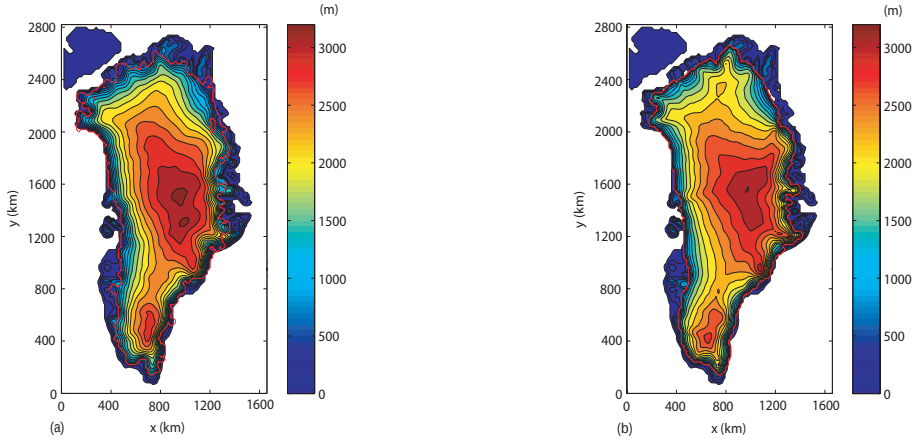


**Fig. 5. (a)** Observed ice-sheet surface elevation (m). **(b)** Steady-state GrIS surface elevation (m) at start of adjoint sensitivity run. Red line is edge of ice-sheet margin.





GMDD
6, 5251–5288, 2013