Geosci. Model Dev., 19, 8269–8288, 2026
https://doi.org/10.5194/gmd-19-8269-2026


Geoscientific Model Development
Open Access
EGU



# A semi-Lagrangian advection scheme in Elmer (v26.1): benchmarking against discontinuous Galerkin and application to ice-damage transport

**Cyrille Mosbeux[1,★], Peter Råback[2,★], Adrien Gilbert[1], Julien Brondex[1], Fabien Gillet-Chaulet[1], Nicolas C. Jourdain[1], Mondher Chekki[1], Olivier Gagliardini[1], and Gaël Durand[1]**

[1]Univ. Grenoble Alpes, CNRS, IRD, Grenoble INP, INRAE, IGE, 38000 Grenoble, France
[2]CSC-IT Center for Science, Espoo, Finland
★These authors contributed equally to this work.

**Correspondence:** Cyrille Mosbeux (cyrille.mosbeux@univ-grenoble-alpes.fr)

Received: 26 June 2025 – Discussion started: 20 August 2025
Revised: 16 March 2026 – Accepted: 7 April 2026 – Published: 8 September 2026

**Abstract.** Transport processes are of great importance in geophysical applications, including atmospheric, oceanic, and ice flow dynamics. An Eulerian view is commonly adopted in models representing fluid dynamics. In such a framework, transport processes are accounted for by prescribing advection terms within the partial differential equations (PDEs) of the model. Yet, advection terms are prone to cause instabilities in the numerical solution of these equations, notably when using the finite element method with a standard Galerkin approach. Various methods have been developed to overcome these instabilities, but often at the price of spurious artificial diffusion. To avoid such unwanted numerical smoothing, a commonly used technique is the discontinuous Galerkin method, which allows for discontinuous solutions; hence, a better tracking of fine features with steep gradients without relying on artificial diffusion. In this study, we explore an alternative approach that lies in semi-Lagrangian schemes, combining elements of both Eulerian and Lagrangian frameworks by updating particle positions based on the Eulerian velocity field from the previous time step. The method does not rely on explicit artificial diffusion and can accurately capture advection while limiting numerical diffusion. Here, we present a computationally efficient semi-Lagrangian algorithm to track the motion of particles in complex 3D geometries that is suitable for highly parallel computing. We illustrate the accuracy and power of the semi-Lagrangian (SL) algorithm by comparing it to a discontinuous Galerkin (DG) method developed within the open-source multi-physics code Elmer. We show that both the DG and SL methods can provide accurate transport solutions with different sensitivity to resolution. We conclude that, for practical use, the choice between the SL and the DG methods will depend on specific simulation requirements and the trade-off between acceptable diffusion and computational efficiency.

## 1 Introduction

Transport processes play a key role in the field of geoscience fluid dynamics, with applications in various fields such as atmospheric, oceanic, and ice flow modeling. In glaciology and ice sheet modeling, pure transport problems are of major importance when it comes to simulating ice age (e.g. Jouvet et al., 2020; Van Liefferinge and Pattyn, 2013), rock and sediment transport (e.g. Wirbel et al., 2018), snow densification (e.g. Gilbert et al., 2014; Gagliardini and Meyssonnier, 1997), anisotropy (e.g. Gillet-Chaulet et al., 2006) or ice damage evolution (e.g. Sun et al., 2017; Krug et al., 2014; Ranganathan et al., 2025). In numerical models, the transport and evolution of quantities can be solved using different frameworks: Eulerian, Lagrangian, and semi-Lagrangian.

Transport processes in a continuum can be described by the advection (or transport) equation, which governs the spatial and temporal evolution of a scalar or tensor quantity $\mathbf{q}$ under a velocity field $\boldsymbol{u}$

$$\frac{\partial \mathbf{q}}{\partial t} + \boldsymbol{u}\nabla\mathbf{q} = \mathbf{S}, \quad (1)$$

with $\mathbf{S}$ a potential source or sink term. This partial differential equation (PDE) can be expressed in different reference frames.

In an Eulerian framework, the equations are solved on a fixed spatial mesh, observing how $\mathbf{q}$ changes as material flows through control volumes. The numerical discretization and resolution of this equation, such as the Galerkin method widely used in finite element methods (FEMs), often presents stability issues (e.g. Ouardghi et al., 2022). In the context of FEMs, a common solution is to rely on the streamline upwind Petrov–Galerkin (SUPG) method (e.g. Hughes, 1987), which modifies the test functions in the weak formulation, introducing a residual-based term that acts as directional diffusion along streamlines. Theoretically, this diffusion vanishes as the residual approaches zero upon mesh and time-step refinement. In practical simulations, however, the residual remains finite and the term effectively behaves as a form of artificial diffusion, introducing numerical smoothing that can reduce the accuracy of sharp gradients.

Variational multiscale (VMS) stabilization frameworks have been proposed to improve upon classical SUPG formulations by better controlling cross-wind oscillations and enhancing robustness on non-uniform meshes (e.g. Masud and Khurram, 2004; Cheng et al., 2024). While these approaches significantly improve the behavior of stabilized continuous Galerkin (CG) methods in many advection-dominated regimes, they still rely on residual-based modeling and may introduce some degree of numerical diffusion, particularly when sharp fronts or strongly localized features must be preserved.

An alternative approach to improving stability while addressing the limitations of SUPG is the discontinuous Galerkin method (DG) (e.g. Reed and Hill, 1973; Zienkiewicz et al., 2003), which is increasingly used within FEMs (e.g. Brezzi et al., 2004; Kuzmin, 2006, 2010). Unlike the standard Galerkin method, which requires the test functions to be smooth and continuous across element boundaries, this method allows the test functions to be discontinuous at those boundaries. Instead, it uses special conditions, called numerical fluxes, to handle how information passes between elements. These fluxes account for information exchange between elements, maintaining stability and accuracy while the discontinuous nature of the test functions allows for sharper resolution of solution features across interfaces. Treating these discontinuities requires the use of "halo" or "ghost" elements surrounding native elements to receive the solution from one partition to another or on domain boundaries, which results in additional computation and communication volume (Brus et al., 2017). Although the method generally improves the solution compared to CG methods, it still experiences inherent numerical diffusion that arises from the numerical flux formulations used to ensure stability at element interfaces.

Particle-based approaches, which are conceptually rooted in the Lagrangian framework, offer a distinct perspective on transport processes. Instead of solving the continuum equations on a fixed spatial grid, these methods explicitly track discrete particles (or markers) as they move along trajectories defined by the flow field (e.g. Samelson and Wiggins, 2006). Physical quantities are carried by the particles themselves and evolve according to the local flow, thereby avoiding the explicit computation of the advection term on a mesh.

More broadly, Lagrangian and hybrid formulations encompass a wide spectrum of approaches beyond particle tracking, including moving-mesh formulations such as Arbitrary Lagrangian–Eulerian (ALE) methods, coupled Eulerian–Lagrangian (CEL) techniques, and Material Point Method (MPM) formulations. These approaches have demonstrated strong capabilities for simulating fully coupled ice-flow, damage, and fracture processes, including rift propagation and evolving ice–ocean boundaries (e.g. Jiménez et al., 2017; Huth et al., 2021a, b, 2023). Particle-in-cell (PiC) tracer approaches have also been used to investigate crevasse advection and its impact on calving dynamics (e.g. Berg and Bassis, 2022).

Although particle-based formulations can scale efficiently on modern high-performance computing systems (e.g. Macpherson et al., 2009; Ketefian et al., 2016), they may still become computationally demanding in large-scale simulations due to the large number of particles that must be tracked and interpolated at each time step. This overhead is particularly relevant in coupled Eulerian–Lagrangian frameworks, where particle advection and particle–mesh interpolation introduce additional communication and memory costs.

In contrast, the present work focuses on the advection of tracer-like quantities within an Eulerian finite-element ice-flow model, for which semi-Lagrangian methods provide a lightweight alternative that avoids maintaining persistent particle populations while limiting numerical diffusion. The semi-Lagrangian scheme stands between Eulerian and Lagrangian formulations by solving the transport equation along characteristic trajectories computed from the Eulerian velocity field. More precisely, the semi-Lagrangian scheme solves the transport equation along characteristic trajectories obtained by backward integration of the Eulerian velocity field (e.g. Côté and Staniforth, 1988; Mortezazadeh et al., 2024). In the present implementation, these trajectories are approximated using particles initialized at mesh locations. Particles are initialized at each node, element barycenter or integration point of the Eulerian mesh at the current time step. Field values are then interpolated from the Eulerian mesh to the former position of the particles and transported to their position at the current time step.

In this study, we present a semi-Lagrangian (SL) method developed in the Elmer finite element model, a multi-physical simulation software mainly developed by CSC

in Finland (https://research.csc.fi/service/elmer/, last access: 15 June 2026) that can solve a large number of partial differential equations, including models for fluid dynamics. While SL and Discontinuous Galerkin (DG) advection schemes were already present in Elmer, the original SL implementation had several limitations that prevented its reliable use for the advection of active tracers such as damage variables. In this work, we revisit and substantially extend this solver by correcting the treatment of source terms depending on the transported variable, improving particle tracking and boundary handling, and ensuring robust parallel execution (see Appendix A1 for implementation improvements). Using this improved implementation, we perform a systematic and quantitative comparison between the SL and DG advection schemes within Elmer, highlighting their respective strengths and limitations in the transport of sharp tracer features. Finally, using the glaciological extension of Elmer, Elmer/Ice, which can be used to simulate complex ice flows (Gagliardini et al., 2013), we demonstrate the ability of both advection schemes to simulate the transport of ice damage (representations of sub-mesh-scale ice crevasses) in a realistic scenario for which numerical diffusion has often been a limiting factor.

## 2 Methods

In this section, we provide a detailed overview of the SL method, starting with its theoretical foundations, followed by the specific implementation of the particle tracking within Elmer. Additionally, we briefly introduce the DG method, which we use for comparison with the SL method.

### 2.1 Semi-Lagrangian advection

By definition, a Lagrangian description of a system consists of following individual particles along their trajectories as opposed to the classical Eulerian description usually used in Elmer, which focuses on the variation of system variables at fixed locations (the grid points). The SL method is based on a Lagrangian discretization of the transport equation but uses the Eulerian velocity field to determine the particle velocities and, therefore, their trajectories. The velocity and pressure fields are typically solved following Stokes-like equations with Elmer's standard continuous Galerkin formulation with stabilized equal-order linear elements. The resulting velocity field is continuous at element nodes, ensuring smooth trajectories for particle advection. In the absence of source/sink, particles are then simply used to carry advected field from their previous to their current locations. To this end, particles are initialized at the mesh nodes (they can also be initialized at the center of the elements or at integration points) and tracked backward in time following the velocity field $\boldsymbol{u}$. In the present implementation, particles are not persistently stored throughout the entire simulation but are periodically reinitialized from the Eulerian field at a user-defined interval (see Sect. 4 for an example of the impact of this choice). When their previous position is recovered, the value of the advected field is reconstructed from the Eulerian mesh using continuous Galerkin (CG) shape functions and assigned to the corresponding nodes at the current time step. These nodal values are subsequently projected to the integration points when required for source-term and velocity-update evaluations (the principle of the method is illustrated in Fig. 1). This is done by evaluating the following integral:

$$\boldsymbol{r}_{t-\Delta t} = \boldsymbol{r}_t + \int_t^{t-\Delta t} \boldsymbol{u}\,\mathrm{d}t \tag{2}$$

where $\boldsymbol{r}$ is the position vector, $t$ is the current time step at which we want to evaluate the new value and new position of the transported variable $q$, and $t - \Delta t$ is the previous time step. In order to account for the spatial variability of the velocity field in the reconstruction of the particle trajectory, the integral in Eq. (2) is evaluated by dividing the main time step of the simulation $\Delta t$ into a discrete number $N$ of internal time steps $\delta t$ so that:

$$\int_t^{t-\Delta t} \boldsymbol{u}\,\mathrm{d}t = \sum_{i=1}^{N} \boldsymbol{r}_i\,. \tag{3}$$

with $\boldsymbol{r}_i$ the position vector at internal time step $i$ (the red dots in Fig. 1). Note that the value of $N$ is a trade-off between an increasing computation time for large $N$ and inaccuracies in the path description, leading to an inaccurate evaluation of the position $r_{t-\Delta_t}$ and therefore the particle value $q(r_t, t)$, for small $N$. The integral may be evaluated using a first-order explicit scheme or a second-order Runge–Kutta scheme. In the first-order scheme, a quadratic correction of the velocity $\boldsymbol{u}$ accounts for the variation of the velocity field across a particle's path, improving the tracking:

$$\boldsymbol{u} = \boldsymbol{u}(t, \boldsymbol{r}_i) + \frac{1}{2}\nabla\boldsymbol{u}(t, \boldsymbol{r}_i) \cdot \boldsymbol{u}(t, \boldsymbol{r}_i)\,\delta t_i \tag{4}$$

with $\boldsymbol{u}(t, r_i)$ and $\nabla\boldsymbol{u}(t, r_i)$ the velocity and velocity gradient at the current time step $t$ and evaluated in $\boldsymbol{r}_i$, respectively. This evaluation is conducted by determining the location of the particle in the element, solving for the local coordinates, and interpolating from nodal values through shape functions. When using a second-order Runge–Kutta temporal discretization, the method already contains inherent quadratic terms, and the correction is not applied. However, the correction of Eq. (4) is quadratic in the velocity field and can lead to oscillations in the computed field, resulting in non-physical particle positions. When the particles have been advected, the field is evaluated from:

$$q = q(\boldsymbol{r}_t, t) = q(\boldsymbol{r}, t - \Delta t) \tag{5}$$

where the variable $q$ in each node depends on the interpolated value of $q$ at the initial position and the earlier time step.

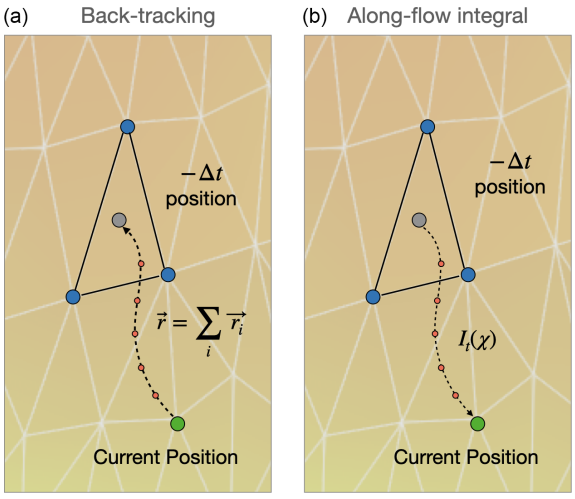


**Figure 1.** Schematics of a finite element mesh and a particle (here a nodal particle in green) that is back-tracked in time until reaching the position it was at previous time step (grey particle with a value interpolated from the finite element). **(a)** The back-tracking is divided in a series of internal time steps $\delta t_i$ that can be adjusted for accuracy. **(b)** The source integral is then evaluated forward in time, over the same number of time steps $\delta t_i$.

A source term $\chi$ that depends on the evolution of the particle along the path integral, following a function $\chi(q,t)$, can be evaluated over time. This corresponds to solving:

$$I_t(\chi) = \int_{t-\Delta t}^{t} \chi(q,t)\,\mathrm{d}t \tag{6}$$

The new value of $q$ is then reevaluated as the combination of Eqs. (5) and (6):

$$q = q(\boldsymbol{r}_{t-\Delta_t}, t) + I_t(\chi) \tag{7}$$

Back-tracked particles have to be localized within the mesh at each internal time step $\delta_i$ to capture the velocity of the particle. The simplest way to find this position is to localize the particle's new location using the coordinate of the particle to retrieve its position into the mesh. The method is accurate but very inefficient for unstructured meshes, making it very greedy in terms of computation time and memory access. In-cell test algorithms have been developed to overcome this issue (e.g. Macpherson et al., 2009; Haselbacher et al., 2007; Ketefian et al., 2016). The in-cell test allows to back-track the mesh cell where the particle resides, without the need for additional searches over the entire mesh. The shape functions associated with the nodes within the element are then evaluated, and the interpolation is performed using the nodal values.

Our particle-tracking algorithm is based on the method developed by Macpherson et al. (2009). It involves tracking the motion of particles from element to element by checking whether or not an element face is crossed over an internal time step. Similarly to Macpherson et al. (2009), let's define a particle with an initial position $\boldsymbol{r}(t)$ corresponding to a node. For each internal time step $\delta_i$ of the back-tracking procedure, a check is performed to evaluate whether or not one of the faces of the element has been crossed. This is done through geometrical considerations via the calculation of determinants between the vectors formed by the initial and final locations of the particle over the internal time step on the one hand, and the vectors formed by the positions of the nodes of the considered element face on the other hand. If it turns out that a face has been crossed, the face index is used to determine which element, if any, is on the other side of the face. Then, the algorithm is continued from this new element for each remaining internal time step $\delta_i$ until the end of the main simulation time step $\Delta_t$. While the method requires some complex geometrical tests, it is shown that computational time scales approximately linearly ($\mathcal{O}(N_p)$) with the number of particles ($N_p$), as only local operations are performed. This makes the method faster than most tree-based localizations that are shown to scale logarithmically ($\mathcal{O}(N_p \log N_p)$). More detail on the particle location and element crossing, as well as the interaction with boundary interfaces can be found in the Appendix A.

### 2.2 Discontinuous Galerkin advection

In Elmer/Ice, the advection equation Eq. (1) can also be solved by applying a discontinuous Galerkin method (DG). Similarly to the SL algorithm, the flow solution does not rely on a DG formulation; only the advection equation employs a DG formulation. The implementation of the method mostly relies on the work of Brezzi et al. (2004) for solving first-order linear hyperbolic equations. In this method, the stability of the numerical scheme (i.e. the stabilization of the oscillations near the discontinuities) is ensured by adding a jump-penalty term to the discretized Eq. (1) without requiring upwind stabilization or other terms. Only a short summary is presented here.

To apply the DG method with jump-penalty stabilization to the advection equation, we begin by rewriting Eq. (1 in its weak form by multiplying it by a discontinuous test function $v_h$ and integrating over each element $T$:

$$\int_T \frac{\partial q_h}{\partial t} v_h \,\mathrm{d}x + \int_T \boldsymbol{u} \cdot \nabla q_h \, v_h \,\mathrm{d}x = \int_T S v_h \,\mathrm{d}x. \tag{8}$$

Integrating the advection term by parts, we obtain:

$$\int_T \boldsymbol{u} \cdot \nabla q_h \, v_h \,\mathrm{d}x = -\int_T q_h \, \boldsymbol{u} \cdot \nabla v_h \,\mathrm{d}x + \int_{\partial T} \boldsymbol{u} \cdot \boldsymbol{n} \, q_h \, v_h \,\mathrm{d}s, \tag{9}$$

where $\boldsymbol{n}$ is the outward normal on $\partial T$.

The DG formulation can then be expressed as:

$$\sum_{T\in\mathcal{T}_h}\int_T \frac{\partial q_h}{\partial t} v_h \,\mathrm{d}x - \sum_{T\in\mathcal{T}_h}\int_T q_h \boldsymbol{u}\cdot\nabla v_h \,\mathrm{d}x + \sum_{e\in\mathcal{E}_h}\int_e \{\boldsymbol{u}\cdot\boldsymbol{n}\, q_h\}[[v_h]]\,\mathrm{d}s + \sum_{e\in\mathcal{E}_h}\int_e \alpha_e [[q_h]][[v_h]]\,\mathrm{d}s = \sum_{T\in\mathcal{T}_h}\int_T S v_h \,\mathrm{d}x. \quad (10)$$

where the third term $\int_e \{\boldsymbol{u}\cdot\boldsymbol{n}\, q_h\}[[v_h]]\,\mathrm{d}s$ captures the advection across the element interface $e$ using the average flux, with $[[v_h]]$ being the jump operator representing the difference in $v_h$ between the two sides of $e$, and $\{\boldsymbol{u}\cdot\boldsymbol{n}\, q_h\}$ represents the mean flux across $e$. The fourth term ensures stability by penalizing large jumps in $q_h$ across element edges, where $\alpha_e$ is a penalty parameter, often set as $\alpha_e = \frac{|\boldsymbol{u}\cdot\boldsymbol{n}|}{2}$. This penalization helps prevent spurious oscillations and improves the convergence of the solution.

## 3 Synthetic experiments

To illustrate the performance of our SL model, we have conducted two numerical experiments that solve simple transport problems. The first experiment, using the Zalesak disc (Zalesak, 1979), allows us to evaluate the ability of our model to transport sharp shapes while guaranteeing their conservation in a simple 2D framework. The second test allows us to evaluate the performance of the transport in a 3D framework. In both of these simple cases, the SL framework is evaluated against the Eulerian advection model usually used in Elmer with a discontinuous-Galerkin (DG) method.

Besides comparing the general patterns, the accuracy of the two methods will be evaluated using two metrics. The first metric assesses the spatial distribution of the advected field and consists of the normalized root mean square error (NRMSE) between a reference solution ($q_{\mathrm{ref}}$) and the advected quantity ($q_i$ with $i$ = SL or DG):

$$\mathrm{NRMSE}(q_i) = \frac{1}{\Omega}\sqrt{\frac{\int_\Omega (q_{\mathrm{ref}} - q_i)^2}{\max(q_{\mathrm{ref}})}}\,\mathrm{d}\Omega. \quad (11)$$

The second metric is used to assess conservation, i.e. we expect the integral of the field $q$ over the domain $\Omega$ to remain constant over time, and the following quantity to remain equal to 0 (for a closed system and without a source/sink term in Eq. 6):

$$\Delta[q_i] = \frac{1}{\Omega}\int_{\Omega_s} \frac{q_i - q_{\mathrm{ref}}}{q_{\mathrm{ref}}}\,\mathrm{d}\Omega. \quad (12)$$

### 3.1 Solid bodies in rotation

This classical rotating-disc experiment is designed to assess the ability of the advection schemes to preserve sharp geometrical features and global mass under purely advective motion. Because the analytical solution corresponds to a rigid-body rotation, any deformation or diffusion of the discs can be directly attributed to numerical errors.

#### 3.1.1 Experimental setup

We define a square domain $\Omega_{\mathrm{square}} = [0,1]\times[0,1]$, with linear (i.e. linear shape functions) square finite elements. We put 2D solid bodies in rotation within a 2D circular steady velocity field with an angular velocity $\omega = 1\,\mathrm{s}^{-1}$. The discs are hence supposed to maintain their shape while rotating, allowing for the evaluation of the advection scheme's ability to accurately transport information. Simulations are performed at coarse ($R_{s,1} = 101\times101$ nodes) and fine resolution ($R_{s,2} = 201\times201$ nodes), and with different time steps, from ($\Delta t = \frac{2\pi}{30}$ s to $\Delta t = \frac{2\pi}{1440}$ s), to assess the sensitivity of the solution to both spatial and temporal resolution.

We perform three types of simulations:

1. SL transport of the discs without reinitialization of the particles, i.e. only one simulation timestep $\Delta t = 2\pi$ s. Given the stationary velocity field, particles do not require to be reinitialized at each time step. In this case, there is almost no loss of information since only one interpolation is needed over the entire simulation – only internal time steps ($\delta_i$) are considered to improve the trajectory of the particles but the interpolation of the field to the particle position is made once at the end of the backward trajectory. As a consequence, we consider that this simulation can be used as a reference for the other methods.

2. SL transport with reinitialization at each time step ($\Delta t < 2\pi$ s). This case shows how the simulation would perform in a transient simulation with $\boldsymbol{u} = \boldsymbol{u}(t)$.

3. Eulerian transport with a DG method. The method is applied in the same conditions as the SL transport to compare the results.

#### 3.1.2 Results

The SL and Eulerian DG methods exhibit opposite sensitivities to spatial and temporal resolution (Figs. 2 and 3). With the SL method, the sharpness of the discs at the end of the solution increases with spatial resolution and decreases with temporal resolution. This is directly linked to interpolation errors: higher spatial resolution improves the interpolation accuracy, while smaller time steps increase the number of interpolations performed during the simulation. As $\Delta t$ decreases, particles move only slightly within the same element and are therefore interpolated more frequently, leading to the accumulation of interpolation errors. For sufficiently small $\Delta t$ (e.g. $\Delta t \sim 2\pi/200$ s for $R_{s1}$ and $\Delta t \sim 2\pi/700$ s for $R_{s2}$), this decrease in accuracy tends to stabilize (Fig. 3a). This saturation occurs because the interpolated values change very

little between successive steps, and the cumulative interpolation error becomes effectively bounded.

On the contrary, the accuracy of the Eulerian DG method increases mostly with temporal resolution, while showing little sensitivity to spatial resolution (Fig. 3a).

In terms of conservation, the SL method leads to a decrease of $[q_i]$ as $\Delta t$ increases, resulting in non-zero values of Eq. (12). This loss can be attributed to the non-conservative nature of the particle–mesh interpolation process, where particle values are reconstructed from nodal values using FEM shape functions. In contrast, with DG, $[q_i]$ remains largely insensitive to changes in $\Delta t$ and is therefore conservative. Finally, the diffusion introduced by the SL method is mostly isotropic, whereas the diffusion in the Eulerian DG scheme is primarily aligned with the velocity field (Fig. 2).

### 3.2 Tracer in a 3D slab flow

This second experiment extends the analysis to a three-dimensional setting and focuses on the influence of time stepping, parallel domain decomposition, and computational cost on tracer advection.

#### 3.2.1 Experimental setup

This experiment involves a 3D parallelepipedal domain with a horizontal surface $\Omega$ of $100\,\mathrm{km} \times 50\,\mathrm{km}$ and a thickness of 100 m. We impose a $1000\,\mathrm{m\,yr^{-1}}$ unidirectional flow along $x$. We use 3D linear hexahedral elements obtained by vertically extruding the 2D rectangular mesh into three layers and conduct the simulations at a coarse resolution ($R_{m,1}$) with $1\,\mathrm{km} \times 1\,\mathrm{km}$ elements, i.e. 15 912 nodes, and a fine resolution ($R_{m,2}$) with $250\,\mathrm{m} \times 250\,\mathrm{m}$ elements, i.e. 401 919 nodes. A donut-shaped tracer is initialized at the position $x = 25\,\mathrm{km}$ and $y = 25\,\mathrm{km}$ (center of the donut) to test the advection algorithms. The simulations are conducted for 50 years with a stationary flow. The horizontal velocities are set up so that the center of the tracer flows from $x = 25\,\mathrm{km}$ to $x = 75\,\mathrm{km}$ over the course of the simulation.

With the increasing availability of high-performance computing systems, most numerical simulations are now executed in parallel. Such parallelization can be challenging for the SL algorithm when one particle moves from one partition to another. We perform the simulations using distributed-memory parallelization with MPI only; no shared-memory (OpenMP) parallelism is employed. Inter-partition communication is carried out after each internal time step of the particle tracking to ensure consistent particle ownership across sub-domain boundaries. A dedicated scalability analysis is presented in Appendix A4. In addition, we compare the computation time of the SL and the DG methods, as well as the impact of the number $N$ of internal time steps $\delta_i$ in Eq. (3).

#### 3.2.2 Results

In this experiment, the SL method shows again a large sensitivity to the time-stepping choice. Given that the grid is regular and the flow velocity is uniform, steady, and unidirectional, the best results are obtained when fixing $\Delta t = m \times \Delta x / v_x$ (with $m \in \mathbb{N}$), which allows a perfect node-to-node displacement at each $\Delta t$ and no interpolation requirement in the absence of any source/sink term. Once we fall into values of $\Delta t < \Delta x / v_x$, we rapidly see a decrease in solution sharpness with increasing diffusion as $\Delta t$ decreases (Fig. 4a). Since the particles follow a straight line, there is no need for internal time steps to improve the accuracy of the trajectory and we see little to no impact of the number of internal time steps $N$ in Eq. (3). Contrary to the previous case, where the solid shapes were moving along both $x$ and $y$ axes, the uni-directional trajectory only leads to interpolation error along $x$, allowing to have no diffusion along $y$, i.e. perpendicularly to the flow.

These results stand again in opposition with the DG method which leads to better performance when using a smaller $\Delta t$ (Fig. 4b), with little impact of the spatial resolution. Focusing on the calculation of the NRMSE (Eq. 11), the SL method yields smaller NRMSE as $\Delta t$ increases, while the DG method leads to higher NRMSE (Fig. 5). In both cases, the rate of the NRMSE evolution decreases over time, which can be due to an increasing numerical accuracy as the solution gets smoother. In terms of concentration, the DG method leads to almost no average concentration loss (i.e. less than 1 % after 50 years) while the SL concentration oscillates up to $\pm 4$ %. This reflects the non-conservative nature of the particle–mesh interpolation: depending on the local configuration, the reconstructed field can introduce either a slight artificial gain or loss when integrated over the domain. The small oscillations visible in Fig. 5 primarily originate from the repeated reinitializations and interpolations of particle values at each simulation time step ($\Delta t$), which leads to the accumulation of rounding and interpolation errors. The number of internal sub-steps ($\delta t$, kept constant here) affects these oscillations only indirectly through its influence on trajectory accuracy. In parallel runs, additional minor deviations may arise from transient particle losses when particles cross sub-domain boundaries between MPI synchronizations; these events are rare and decrease as synchronization frequency increases.

The computation time also differs between the methods. For identical $(\Delta t, \Delta x)$, the computing time of the SL method increases with the number of internal time steps $N$ (see Table 1). This behavior is approximately linear, as interpolation and mesh-related operations are performed once per simulation time step, whereas the particle advection cost scales with $N$. Although large values of $N$ are not required in the present configuration, they increase the SL runtime relative to the DG method, whose cost typically falls within the range of the SL solver for $N = 3$. Nevertheless, for the configurations con-

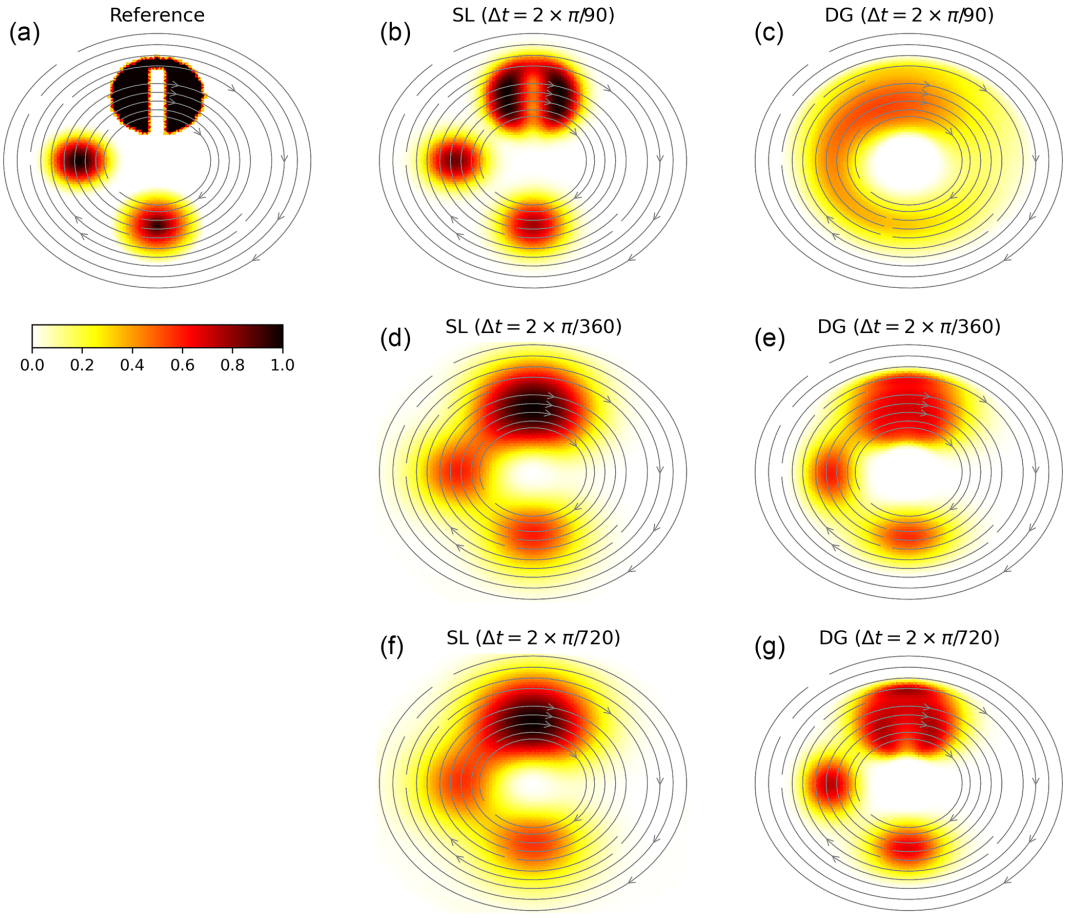


**Figure 2.** Evolution of two discs and one Zalesak disc after a complete rotation: **(a)** Reference solution (i.e. the SL solution after one rotation with no reinitialisation), **(b, d, f)** SL solution and **(c, e, g)** Eulerian DG. The simulation is conducted at the resolution $R_{s1}$ with 3 different $\Delta t$: (first line) $\Delta t = 2 \times \pi/90$, (second line) $\Delta t = 2 \times \pi/360$, (third line) $\Delta t = 2 \times \pi/720$. The grey arrows show the circular direction of the flow.

sidered here, these advection costs remain small compared with the computational expense of the 3D flow solvers (e.g. Stokes flow) typically used in Elmer.

## 4 Application of semi-Lagrangian and discontinuous Galerkin advection to ice-damage evolution in Antarctica

The experiments conducted to assess the performance of the SL method in comparison with the DG method provide valuable insights into the behavior of these numerical schemes under different conditions. These results demonstrate the trade-offs involved in using SL and DG methods, particularly in terms of accuracy, computational efficiency, and sensitivity to spatial and temporal resolutions.

Hereafter, we apply both advection schemes to a high-resolution ice sheet model to simulate the evolution of ice damage in the Amundsen Sea region, where a rapid and significant ice-sheet mass loss has been observed over the last decades (e.g. Smith et al., 2020). This rapid mass loss has led to structural changes in the ice sheet and increased damage in key areas such as shear margins (Sun and Gudmundsson, 2023; Lhermitte et al., 2020; Alley et al., 2019). These damaged areas correspond to highly crevassed regions that often appear where the ice becomes afloat and evolve as they are advected downstream over the ocean.

### 4.1 Numerical ice sheet setup and experiment

We build a 3D model of the region and simulate the ice flow using the state-of-the-art Stokes flow model Elmer/Ice (see Appendix B1), following the initialization procedure described in Appendix B2. The model is discretized on an unstructured finite-element mesh using stabilized equal-order linear wedge elements for velocity and pressure. Numerical integration is performed using 44 Gauss points per element (11 points on the triangular face and 4 along the vertical direction). Although the velocity and pressure are approximated with first-order Lagrange elements, this relatively high

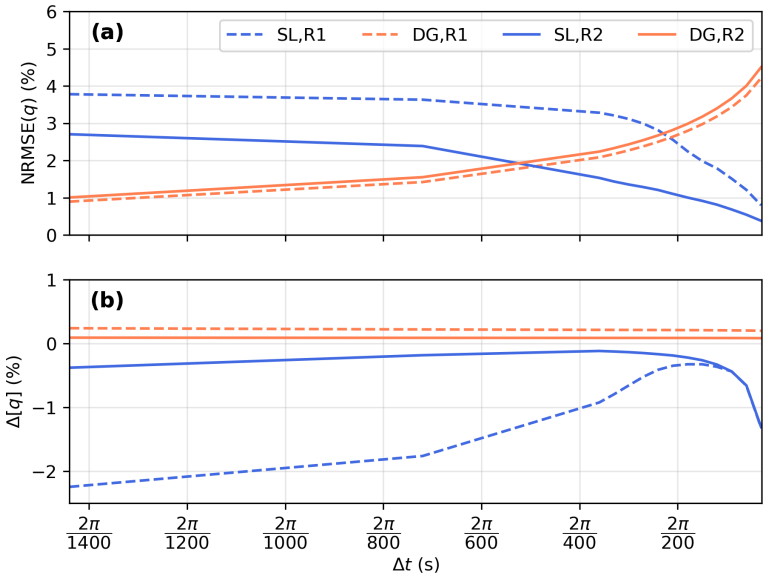


**Figure 3.** Assessment of the SL and Eulerian DG methods after a full rotation of the three solid bodies: **(a)** NRMSE (Eq. 11) and **(b)** spatially-integrated relative bias in concentration (Eq. 12). The results are shown for low ($R_{s1} = 101 \times 101$ nodes) and high ($R_{s2} = 201 \times 201$ nodes) spatial resolutions.

**Table 1.** Average time (in seconds) spent in the solver over one simulation time step ($\Delta t$) for different algorithms for the 3D-slab-flow case of Sect. 3.2 for an 8-partition domain and a 1 km resolution.

| Method | DG | SL ($N = 1$) | SL ($N = 5$) | SL ($N = 10$) |
|---|---|---|---|---|
| Average time step time (s) | 0.23 | 0.11 | 0.41 | 0.78 |

quadrature order is required to accurately integrate the nonlinear viscosity and stabilization terms within each element. Stresses are computed at these integration points and subsequently interpolated to the mesh nodes, where the damage variable is evaluated. Once initialized, we simulate the evolution of damage over 50 years.

Our damage model is based on Continuous Damage Mechanics (CDM) and follows the same physical approach as Krug et al. (2014) and Gilbert et al. (2015): damage is created where the maximum tensile principal stress exceeds a threshold and is advected downstream with the flow. The corresponding advection equation with a source term is solved using either the SL or the DG solver described previously.

To keep the case as simple as possible, we use a constant source term for damage and focus only on the creation of surface damage (i.e. we do not include basal damage due to water pressure in Eq. B11). We also ignore feedbacks between damage and viscosity (see Eq. B6). Details of the damage model are presented in Appendix B3. A key parameter of the model is the stress threshold $\sigma_{\text{th}}$ at which damage occurs. In previous studies, $\sigma_{\text{th}}$ is often taken between 0.1–0.3 MPa (e.g. Krug et al., 2014; Albrecht and Levermann, 2012; Grinsted et al., 2024). Here, we select a particularly high value of $\sigma_{\text{th}} = 0.55$ MPa to limit damage production ($\chi > 0$) to areas clearly identified as crevasse onset regions in observations (Lhermitte et al., 2020). Additionally, mountainous regions and steep slopes are excluded from the simulations as these areas are unlikely to sustain ice cover.

Our synthetic experiments indicated that using longer SL advection time steps ($\Delta t$) combined with multiple internal time steps ($\delta_i$) helps limit the interpolation errors that accumulate over repeated advection cycles. This issue is particularly acute in high-resolution ice-flow models (e.g. $\Delta x \sim 1$ km), where small time steps (e.g. $\Delta t \sim 10^{-2}$ year) are typically required for the flow solver to limit feedbacks between vertical velocities and free-surface evolution.

To address this, the SL advection step is executed with a larger time step than the flow solver ($\Delta t > \Delta t_{\text{flow}}$), reducing both interpolation frequency and computational cost. Because feedbacks between damage and viscosity are disabled in this experiment, this reduced update frequency does not affect the flow solution. In a fully coupled setup, such decoupling would introduce a trade-off between numerical diffusion and the accuracy of damage-flow interactions. However, as our focus is solely on evaluating numerical diffusion, deactivating the feedback allows us to better compare stepping choices.

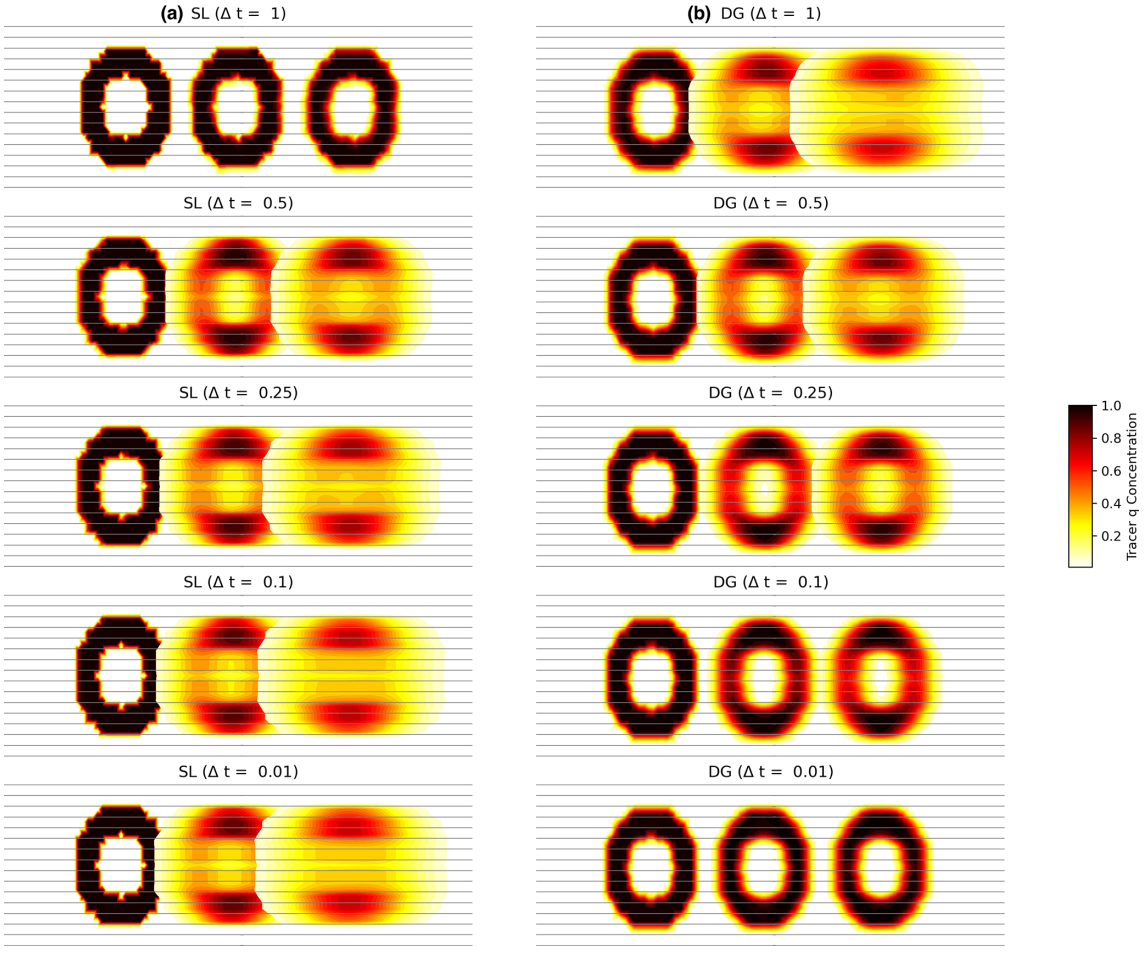


**Figure 4.** Donut transport with the **(a)** SL method (with $N = 5$) and **(b)** the DG method, at 1 km resolution. The donut advects from left to right and is plotted on each panel at $t = 0$ yr, $t = 25$ yr, and $t = 50$ yr. Flowlines are represented in grey.

We conduct a set of eight simulations of damage evolution using the same steady-state geometry, ice flow, and damage model parameters. Each simulation covers 50 years of damage evolution with SL advection time steps ranging from $\Delta t = 10^{-2}$ to 5 year. The same simulations are performed using the DG advection scheme for comparison.

### 4.2 Damage simulation results

The simulations reproduce the main damage structures observed in the Amundsen Sea sector, with damage primarily generated in shear margins near the grounding line of Pine Island Glacier and subsequently advected downstream along flowlines that follow patterns observed in satellite imagery (e.g. Lhermitte et al., 2020). Over the Thwaites Ice Shelf, damage is more widespread but still aligns with the general patterns inferred from observations. Both advection schemes capture the large-scale transport of damage from upstream source regions toward the ice shelves.

In a purely advective setting without numerical diffusion, damage generated in localized source regions should remain laterally confined to the downstream flowlines along which it originates. To visualize potential spreading caused by numerical diffusion, we overlay flowlines with spacing ranging from about one to three elements (approximately 500 m to 5 km depending on the region).

For the SL simulations, decreasing the advection time step $\Delta t$–and therefore increasing the number of particle–mesh interpolation operations–leads to increased smoothing of the vertically integrated damage field, producing progressively smoother damage patterns (Fig. 6). This effect is particularly noticeable over Thwaites Glacier and its ice shelf, where margins and sharp structures become increasingly diffuse as $\Delta t$ decreases. For instance, for $\Delta t = 1$ yr (Fig. 6a), only limited lateral spreading is observed: damage generated upstream remains largely confined within one flowline spacing ($D_{\text{mean}} < 10^{-3}$). Smaller time steps of $\Delta t = 10^{-1}$ yr (Fig. 6b) and $\Delta t = 10^{-2}$ yr (Fig. 6c) lead to progressively stronger lateral spreading, with damage originating from

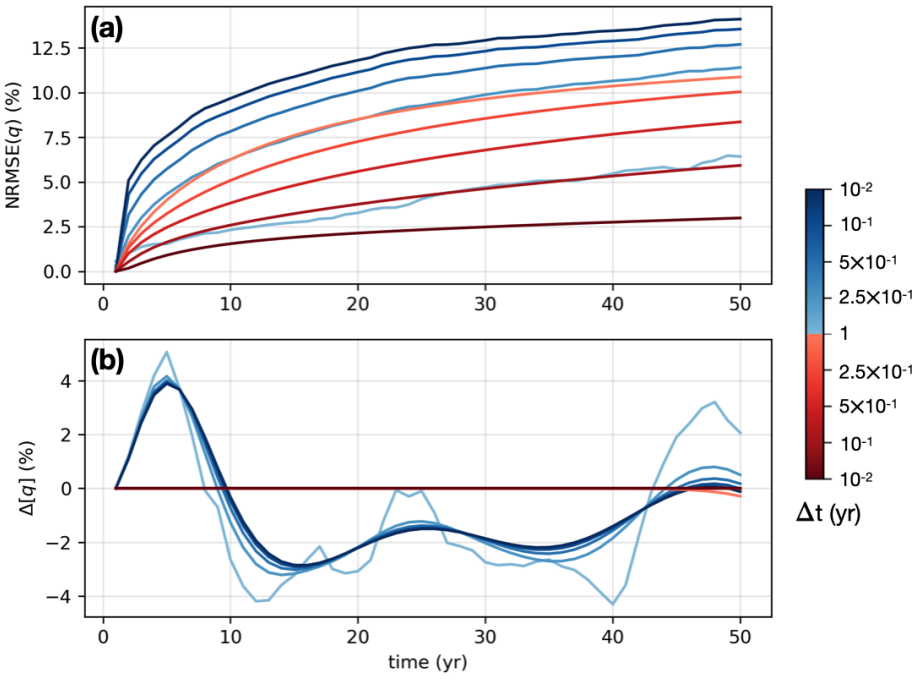


**Figure 5.** **(a)** NRMSE and **(b)** spatially-integrated relative bias in concentration of the donut tracer over time for different time step lengths at 1 km resolution (SL: blue shades; DG: red shades).

neighboring ice streams occasionally merging downstream. In addition, we observe cumulative smoothing and a gradual loss of damage intensity along the flowlines as the signal is advected farther from its source, an effect that is particularly visible along the shear margins of Pine Island Glacier and consistent with the non-conservative nature of the particle–mesh remapping.

The DG simulations exhibit a different spreading structure. While the large-scale damage patterns remain comparable, diffusion occurs preferentially along the flow direction. As a result, damage features tend to extend farther downstream from the source regions for larger time steps, producing elongated streaks aligned with the velocity field. This contrasts with the SL solution, where smoothing arises primarily from particle–mesh interpolation and therefore appears more isotropic, with only a secondary influence from mesh orientation. At the same time, the DG formulation preserves tracer intensity more effectively, reflecting its locally conservative nature. These differences are consistent with the behavior observed in the synthetic benchmark experiments. Small oscillatory patterns can also be observed along some DG damage trajectories, appearing as triangular structures aligned with the mesh elements. These features likely reflect element-wise reconstruction and limiting operations used in the DG discretization. Although these mesh-scale artefacts remain localized and do not significantly affect the large-scale damage distribution, they highlight differences in how the two schemes handle steep tracer gradients during advection.

We can also assess numerical diffusion by examining three vertical damage profiles at different locations (see locations in Fig. 6 and profiles in Fig. 8). On *Profile 1* (Fig. 8)a, d), located within a shear margin of Pine Island Glacier and downstream of a damage source, the overall damage distribution remains relatively similar for both advection schemes. However, for the SL simulations, decreasing $\Delta t$ leads to a reduction in the maximum damage near the surface. The DG simulations show the opposite tendency, with the surface maximum decreasing as $\Delta t$ increases. We also observe a deeper penetration of damage into the ice column as $\Delta t$ decreases or increases, depending on the method but this effect is stronger with SL. One exception occurs for very large SL time steps (e.g. $\Delta t = 5$ yr), which produce slightly lower damage within the upper $\sim 100$ m of the ice column. This behavior may result from reduced numerical diffusion of damage originating from adjacent flowlines. *Profile 2*, located downstream of a damage source of Thwaites Glacier, exhibits a similar pattern (Fig. 8b and e). Conversely, *Profile 3* which is situated between two damage sources and is not downstream of any specific source, exhibits low to no damage for $\Delta t = 5$ yr SL simulation. In the absence of numerical diffusion, the ice should remain undamaged at this location. However, due to numerical diffusion, damage appears and increases rapidly as $\Delta t$ decreases for the SL, before stabilizing once the particle displacement per time step becomes small enough that further interpolations have a lesser impact on the results (Fig. 8c). The DG simulations also show the effect of numerical diffusion but to a lesser extent. Overall, the vertical profiles

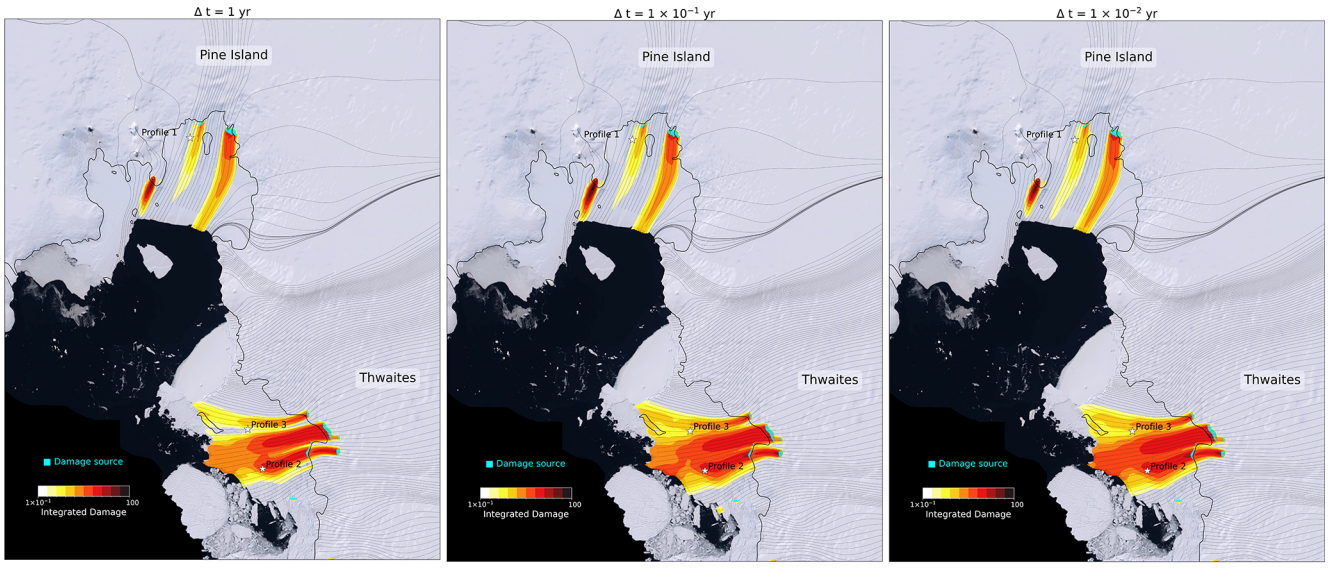


**Figure 6.** Semi-Lagrangian (SL) vertically integrated damage after 50 years of simulation in the Amundsen Sea Sector for different time steps: **(a)** $\Delta t = 1$ yr, **(b)** $\Delta t = 10^{-1}$ yr, and **(c)** $\Delta t = 10^{-2}$ yr. All the simulations were conducted with a $\sigma_{\text{th}} = 0.55$ MPa, and no damage retro-action neither on the viscosity nor on the damage source term, i.e. $D = 0$ in Eqs. (B6) and (B11). The grounding line is represented with a thick black line, the flowlines with thin grey lines, and the sources for damage are represented in cyan. The Landsat Image Mosaic of Antarctica (LIMA) for the region is plotted in background. The location of Pine Island and Thwaites glacier is indicated for reference.

for both methods tend to converge toward similar damage distributions as $\Delta t$ decreases for SL and increases for DG, suggesting that numerical diffusion approaches a plateau for both schemes

In addition to the differences in numerical diffusion, the two advection schemes exhibit markedly different computational costs in the present configuration. In the Antarctic experiments, the DG simulations require an advection time step of order $\Delta t \approx 10^{-2}$ yr. In contrast, accurate SL solutions are obtained with $\Delta t = 1$ yr using $N = 10$ internal timesteps for trajectory integration. Under these conditions, the SL advection step is roughly $\sim 50$–$100$ times cheaper than the DG solver at the same mesh resolution. While the SL still shows diffusion at $\Delta t = 1$ yr, this computing time difference could allow for the SL solver to run with $\sim 50$–$100$ times more particles for a similar cost as the DG solver at $\Delta t \approx 10^{-2}$ yr.

## 5 Discussion and future adaptations

The current application of the SL method to damage evolution in the Amundsen Sea sector of Antarctica provides a practical demonstration of both the challenges and benefits of the method in glaciological simulations. The SL method's sensitivity to time step length directly impacts the diffusion of the damage field: longer time steps reduce numerical diffusion. The observed lateral diffusion of damage along flowlines, especially in regions like Thwaites Glacier, underscores the importance of carefully selecting time step parameters to minimize unwanted diffusion while ensuring computational feasibility. While small time steps are typically required for solving ice flow (i.e. Stokes flow in our case) and free-surface evolution (usually about $\Delta t = 10^{-2}$ yr), we have proposed to address the issue by decoupling the time stepping of damage evolution from the rest of the simulation. However, this approach may compromise the accuracy of the coupling between ice flow and damage evolution, which could pose challenges in capturing short timescale dynamics, such as sub-annual variations in ice flow and damage. More generally, this highlights that DG and SL schemes exhibit complementary numerical behaviors that are controlled by different error mechanisms. DG primarily benefits from time-step refinement, whereas the SL accuracy is largely controlled by spatial resolution and, critically, by the number of interpolation operations: fewer (but larger) advection steps reduce the accumulation of interpolation-driven diffusion. This distinction is important in coupled applications, where frequent exchanges between the flow solver and the transport step can amplify interpolation errors and provide the motivation for testing reduced coupling frequencies in Sect. 4.

Another practical advantage of the SL formulation is its ability to advect multiple fields within a single solver execution. This feature makes the method particularly attractive for problems involving several passively advected quantities, such as tracers, since particle trajectories can be computed once and reused for all transported variables. When source or sink terms are present, only the trajectories can be shared, while the source and sink contributions must still be evaluated separately for each field. Nevertheless, this strategy can substantially reduce the computational cost compared to Eu-

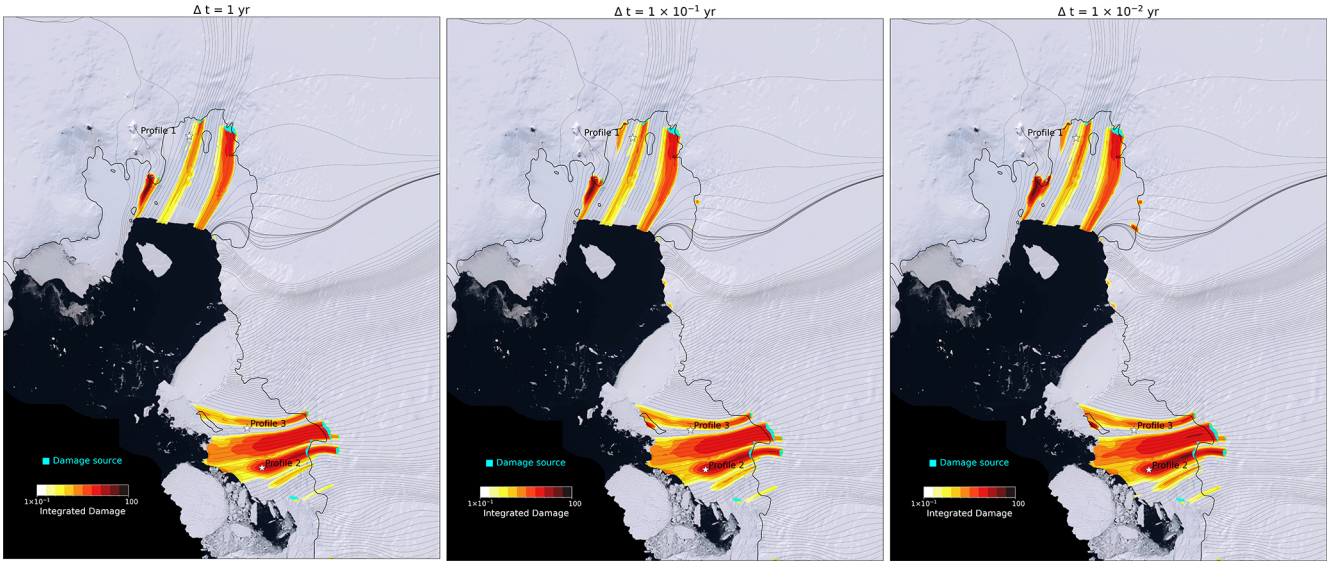


**Figure 7.** Same as Fig. 6 but with the Discontinuous Galerkin (DG) solver. The Landsat Image Mosaic of Antarctica (LIMA) for the region is plotted in background.

lerian approaches, which typically require solving the full advection equation independently for each transported variable.

We outline four strategies to further improve the accuracy of the SL method in Elmer and potentially increase the coupling frequency between Stokes flow and SL damage advection, without introducing additional numerical diffusion:

- Since the additional computing cost of the SL method is usually relatively cheap to run with respect to the Stokes flow (or any flow) simulation, one possible improvement would be to run the SL method on a higher-resolution mesh than the rest of the simulation. In this approach, the Stokes flow would be computed on a coarser base mesh, and the resulting velocity field and SL source term would be interpolated onto the finer mesh for the advection step. At each time step, the advected field – such as damage – would be updated on the fine mesh and then reinterpolated back onto the coarse mesh to compute a new viscosity and solve the Stokes problem again. The velocity field will remain unchanged across both meshes but the finer resolution and the increased number of particles will improve the interpolation of the advected fields.

- Another solution would consist in dynamically adjusting the mesh during the simulation, enabling local refinement in regions of strong gradients and thereby reducing interpolation errors. Adaptive mesh refinement strategies have been shown to improve the representation of sharp advective features by locally increasing resolution (Wirbel et al., 2018). Remeshing tools such as Dapogny et al. (2014) are already interfaced with Elmer but their impact in the present context still needs to be quantified, in particular with respect to the trade-off between accuracy gains, additional computational cost, and potential interpolation errors introduced by the remeshing procedure itself.

- Instead of increasing the resolution, the SL particles could be initialized at Gauss points used for the integration of the flow problem, which number can be larger than the number of nodes (Ouardghi et al., 2022, e.g.). For example, in Elmer, we typically employ up to 44 Gauss integration points when solving the Stokes equations on linear triangular wedge elements, which could considerably increase the number of particles and the accuracy of trajectories and interpolation. However, this solution requires additional development to our current implementation of the SL problem in Elmer.

- Increasing the spatial resolution of the model can reduce spatially integrated bias in the concentration of the solution, as demonstrated in our first experiment. However, a more robust solution would be to develop a conservative formulation for the particle–mesh remapping, which is not currently implemented in Elmer. The synthetic tracer experiments indicate that the dominant limitation of the present SL implementation is not the accuracy of the trajectory integration itself, but the repeated interpolation used to reconstruct the advected field on the Eulerian mesh. Because this remapping is not strictly conservative, it introduces cumulative smoothing and a gradual loss of tracer intensity over time. Conservative SL approaches enforce tracer conservation during the remapping step by redistributing tracer mass across the computational mesh, and would

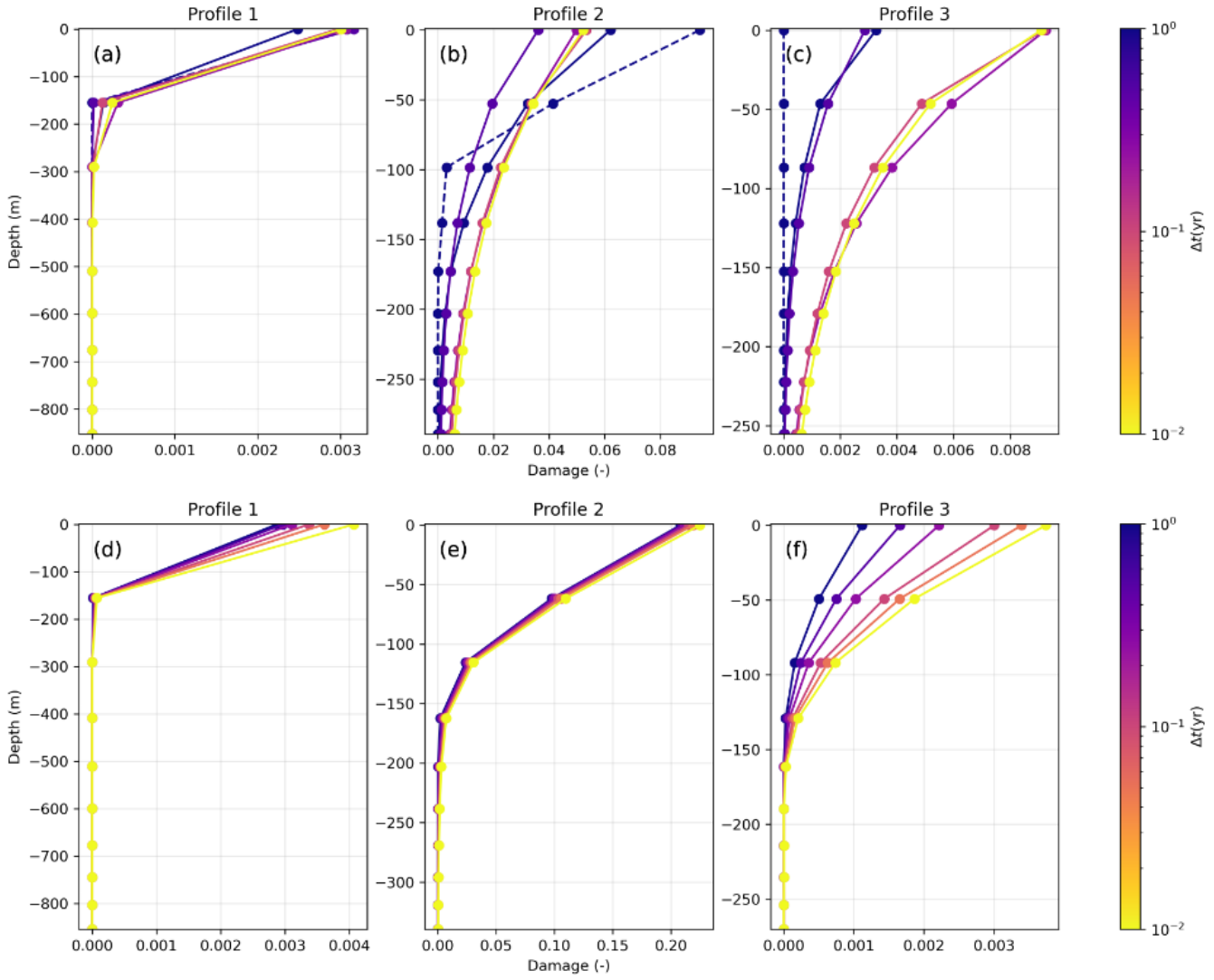


**Figure 8.** Vertical damage profiles after 50 years of simulation for three locations shown in Figs. 6 and 7. Panels **(a–c)** correspond to the semi-Lagrangian (SL) solver and panels **(d–f)** to the Discontinuous Galerkin (DG) solver. The three columns show results for **(a, d)** *Profile 1*, **(b, e)** *Profile 2* **(b, e)**, and **(c, f)** *Profile 3*. Colored solid lines indicate the different advection time steps $\Delta t$, while the dashed line indicates the case $\Delta t = 5$ yr.

likely improve amplitude preservation in the present framework. However, conservative remapping schemes can also introduce shape distortions or numerical oscillations due to the inherent difficulty of simultaneously preserving conservation and monotonicity (e.g. Priestley, 1993).

Taken together, the experiments indicate that the main limitation of the present SL implementation lies in the non-conservative particle–mesh remapping used to reconstruct the advected field. While the DG formulation better preserves tracer intensity, it introduces diffusion preferentially aligned with the flow direction and remains, to some extent constrained by smaller timesteps.

Developing conservative remapping strategies that better preserve tracer amplitude while maintaining the flexibility of the SL approach, therefore, represents an important direction for future developments in Elmer.

Finally, particle–grid approaches such as Particle-in-Cell (PiC) or Markers-in-Cell, introduced in Sect. 1, could provide an alternative strategy for tracer advection. In these methods, particles are advected forward in time while carrying tracer properties, and their values are periodically projected back onto the Eulerian mesh using conservative weighting, which can improve mass conservation and represent sub-grid variability. However, such schemes typically require maintaining a large number of particles per element to ensure adequate sampling, and particle–mesh communi-

cation as well as particle repopulation can become costly on unstructured meshes. While implementing an efficient PiC framework in Elmer would require substantial modifications to the current FEM-based data structures, they represent a promising avenue for the future evolution of Elmer particle-based implementations.

## 6 Conclusions

In this study, we implemented a semi-Lagrangian (SL) advection scheme in the finite-element ice-flow model Elmer/Ice (v26.1) and evaluated its performance for the transport of ice-damage fields. The SL formulation was benchmarked against a discontinuous Galerkin (DG) advection scheme through idealized tests and applied to a realistic Antarctic configuration to assess its behavior in a glaciological context. The results highlight the distinct numerical characteristics of the two approaches: the SL method enables the use of larger time steps with competitive accuracy but introduces numerical diffusion associated with repeated particle–mesh interpolation, whereas the DG scheme provides sharper transport of damage gradients at the cost of stricter time-step constraints. The Antarctic application demonstrates that both approaches can be used to simulate the large-scale evolution of ice damage, while illustrating the trade-offs between computational efficiency and numerical diffusion when modeling damage transport in ice-sheet simulations.

The DG method is particularly effective in scenarios requiring high temporal precision, as it shows significant accuracy improvements with smaller time steps. This makes DG suitable for rapidly evolving flows. On the other hand, the SL method benefits from increasing the time step size and spatial resolution, performing best when the time step allows particles to move precisely from node to node. This characteristic makes the SL method advantageous in relatively steady flows where the time step at which the transport equation is solved can be decoupled from the time step at which the velocity field is recomputed.

In recent years, research on ice sheet and glacier evolution has highlighted the importance of damage processes, but without thoroughly addressing the limitations of current advection methods regarding numerical stability, potential artifacts, or excessive diffusion (e.g. Sun et al., 2017; Lhermitte et al., 2020; Ranganathan et al., 2025; Li et al., 2025). The application of the SL method to ice damage evolution in the Amundsen Sea sector yields promising results, suggesting that Elmer/Ice simulations could effectively incorporate ice damage–a critical factor for accurate predictions of ice sheet and glacier evolution. Although the present simulations are steady in order to facilitate the characterization of numerical diffusion, the formulation of the SL advection scheme does not introduce mechanisms that would inherently lead to convergence or stability issues in transient simulations.

Looking ahead, the SL method's potential is likely to grow with advancements in computing power, which will enable finer spatial resolutions but also longer stable time steps of the free-surface problem in ice-flow and ice-sheet simulations (e.g. Löfgren et al., 2022).

For now, Elmer allows the SL method to be run at a lower frequency than the flow model itself, reducing diffusion at the expense of physical precision. As computing capabilities advance, we expect the SL method to become an even more powerful tool in glaciological simulations.

## Appendix A: Semi-Lagrangian Model

### A1 Implementation improvements to the semi-Lagrangian solver in Elmer/Ice

The SL solver used in this study has existed in the Elmer framework for many years. However, several limitations in the original implementation made it unsuitable for the advection of active tracers such as the damage variable considered here. In particular, the previous formulation did not properly handle source terms that depend explicitly on the transported variable. As part of the present work, several improvements and corrections have been implemented and validated. These include:

- Correct treatment of source terms that explicitly depend on the advected variable.
- Improvement of the path-integral formulation by allowing forward integration of the source term along the particle trajectory, as illustrated schematically in Fig. 1.
- Improved handling of initial conditions, enabling reliable restart capabilities for long simulations.
- Improved treatment of boundary conditions to avoid undesired particle loss near domain boundaries.
- Corrections and improvements related to parallel execution and MPI communication.

These developments were implemented primarily in the modules `ParticleUtils.F90` and `ParticleAdvector.F90`. The evolution of these implementations is documented in the Elmer GitHub repository. The improvements were validated through extensive testing in both idealized benchmark experiments and realistic glaciological simulations.

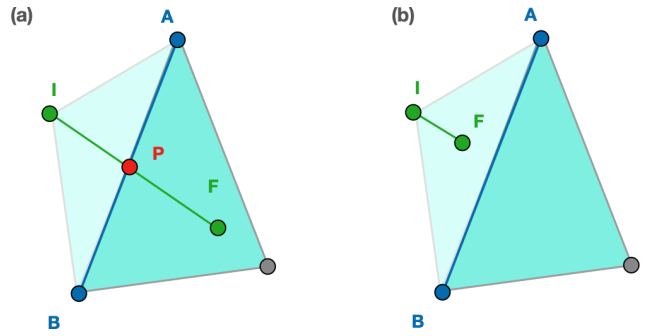


**Figure A1.** Representation of a trajectory ($\boldsymbol{IF}$) from an initial point ($I$, here at a node of a triangular element) to a final point $F$ in another element. **(a)** The segment $IF$ intersects the face segment $AB$ at point $P$. **(b)** The segment $IF$ is too short to intersect the face segment $AB$.

## A2 Particle location and element crossing

The first part of the motion (in the starting element and until reaching the element face at a location $\boldsymbol{p}$) can be found using:

$$\boldsymbol{p} = \boldsymbol{r}(t) + \lambda(\boldsymbol{r}(t-\delta t) - \boldsymbol{r}(t)) \tag{A1}$$

where $\lambda$ is the fraction along the line $\boldsymbol{r}(t-\delta t) - \boldsymbol{r}(t)$ where the interaction occurs with the element face.

In the case of a 2D plan tracking, we can define $I$ as the point corresponding to the initial position (i.e. at $t$) of the particle, $F$ as the point corresponding to the final position (i.e. at $t-\delta t$) of the particle, and $A$ and $B$ as the positions of the two nodes forming the tested face (or segment in this 2D case; Fig. A1).

Let's define two vectors, $\boldsymbol{IF}$ and $\boldsymbol{AB}$. If $\boldsymbol{IF} \times \boldsymbol{AB} = 0$, this indicates that $\boldsymbol{IF}$ and $\boldsymbol{AB}$ are parallel, and therefore, no intersection exists between the lines they define.

Next, we compute $\frac{\boldsymbol{IF} \times \boldsymbol{IA}}{\boldsymbol{IF} \times \boldsymbol{AB}}$. If this ratio is negative or greater than 1, it indicates that the line defined by $\boldsymbol{IF}$ intersects the line defined by $\boldsymbol{AB}$ outside the segment $AB$. As a result, there is no intersection between the particle path and the segment $AB$.

Finally, we calculate $\lambda$ following

$$\lambda = \frac{\boldsymbol{IA} \times \boldsymbol{AB}}{\boldsymbol{IF} \times \boldsymbol{AB}}. \tag{A2}$$

If $\lambda > 1$, the segment $IF$ is too short to intersect $AB$. If $\lambda < 0$, there is also no intersection (point $I$ is on the opposite side of $AB$). If $\lambda < 1$, an intersection occurs, where

$$\lambda = \frac{IP}{IF}, \tag{A3}$$

with $P$ representing the intersection point of lines $IF$ and $AB$.

## A3 Boundary interactions and parallelization

At every face crossing, a check is performed to determine whether the face corresponds to a domain boundary or is an interface between two elements of the mesh. In case of parallel computing, the element face crossed can also stand at the interface between two mesh partitions. Specific actions can be undertaken depending on the type of boundary encountered:

- partition boundary: the particle position is moved to the other partition and the tracking continues in this partition.
- physical boundary: the particle is back-tracked outside the mesh. This can happen when the velocity vector points outward across the domain boundary, i.e. is not tangent to the boundary. In such case, two options are possible: (1) the particle tracking can be stopped at the boundary, effectively treating it as a solid wall; or (2) the particle can be allowed to continue its trajectory along a path tangent to the boundary surface. The same occurs for the forward tracking of the source term.

## A4 Scalability Analysis

Scaling tests were conducted on the 3D slab flow experiment with the SL method (Sect. 3.2). Strong-scaling tests were performed by fixing the global problem size while increasing the number of MPI partitions from 1 to 256. Weak-scaling tests were conducted by increasing the global problem size proportionally to the number of partitions while keeping the number of mesh nodes and particles per partition approximately constant (with less than 1 % variation in the number of nodes per partition). All experiments were executed using distributed-memory MPI parallelization only on a high-performance computing system equipped with dual AMD EPYC Rome processors (128 cores per node) and 228 GB of RAM per node (approximately 1.8 GB per core).

Strong-scaling efficiency remains above 0.8 up to 16 partitions and decreases to approximately 43 % at 128 partitions, reflecting the growing impact of communication and synchronization costs relative to computation. The solver exhibits good weak-scaling behavior up to 64 partitions, with efficiencies above 80 %. At larger partition counts, efficiency decreases gradually due to increased inter-partition particle exchanges and MPI communication overhead. Nevertheless, the corresponding wall-clock time increase remains moderate, indicating acceptable scalability for large three-dimensional applications.

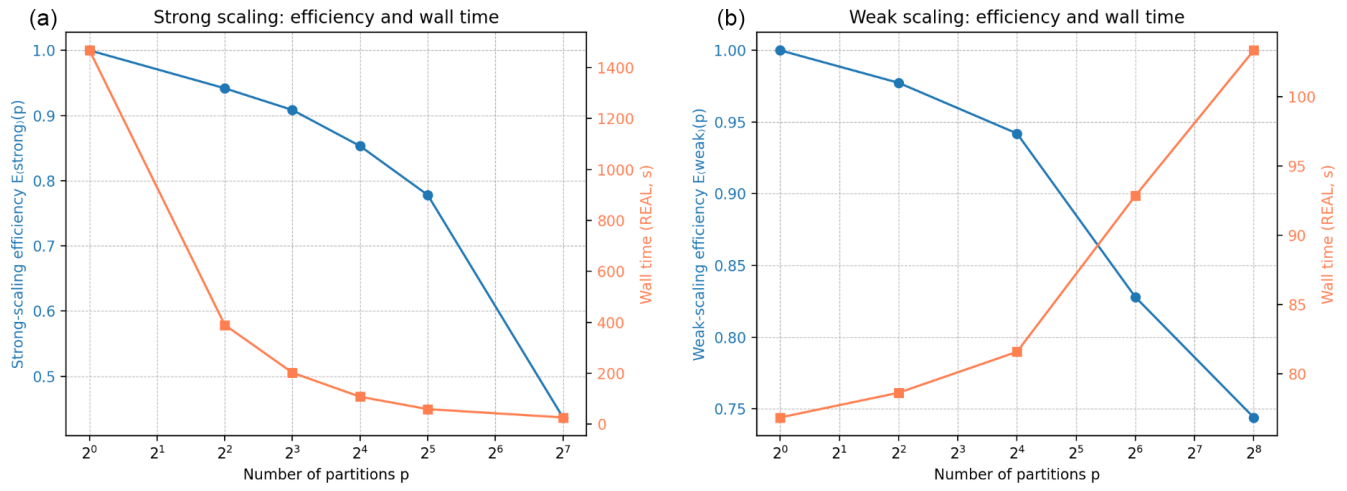


**Figure A2.** Parallel scalability of the semi-Lagrangian solver. **(a)** Strong scaling efficiency and wall-clock time as a function of the number of partitions $p$, obtained by increasing the number of processors while keeping the total problem size constant. **(b)** Weak scaling efficiency and wall-clock time as a function of $p$, where the workload per partition is kept constant. Efficiency is defined relative to the single-partition reference case, and wall time corresponds to the elapsed time for 100 simulation time steps to reduce variability.

## Appendix B: Mechanical Models

### B1 Ice flow model

For continuously deforming fluids, the equation of state can be written as:

$$\nabla \cdot \boldsymbol{\sigma} + \rho_i \boldsymbol{g} = 0, \tag{B1}$$

where $\nabla\cdot$ is the divergence operator, $\boldsymbol{g}$ is gravity, and $\boldsymbol{\sigma} = \boldsymbol{\tau} - p\mathbf{I}$ is the Cauchy stress tensor, which links the deviatoric stress tensor $\boldsymbol{\tau}$ and the isotropic pressure $p$, where $\mathbf{I}$ is the identity matrix. Stress can then be linked to strain rate with an isotropic power law known as Glen's flow law (Glen, 1955), written as

$$\boldsymbol{\tau} = 2\eta\dot{\boldsymbol{\varepsilon}}, \tag{B2}$$

where $\eta$ is the effective viscosity, and $\dot{\boldsymbol{\varepsilon}}$ is the strain-rate tensor, defined as

$$\dot{\varepsilon}_{ij} \equiv \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right). \tag{B3}$$

with $u_i$ the components of the velocity vector $\boldsymbol{u}$. The effective viscosity, $\eta$, in Eq. (B2) is given by:

$$\eta = \frac{1}{2}(E_{\mathrm{D}}A)^{-1/n}\boldsymbol{I}_{\dot{\boldsymbol{\varepsilon}}2}^{(1-n)/n}, \tag{B4}$$

where $E_{\mathrm{D}}$ is an enhancement factor, $\boldsymbol{I}_{\dot{\boldsymbol{\varepsilon}}2} = \sqrt{\dot{\boldsymbol{\varepsilon}}_{ij}\dot{\boldsymbol{\varepsilon}}_{ij}/2}$ is the second invariant of the strain-rate tensor and $n$ is the Glen exponent, with an empirically determined value between 1–5 (Weertman, 1983; Gillet-Chaulet et al., 2011). An average value of $n = 3$ is usually used in ice sheet models and is also applied in this study. $A$ is the fluidity, which depends on the temperature following Arrhenius' law:

$$A = A_0 \mathrm{e}^{-Q/(RT)}, \tag{B5}$$

with $A_0$ a reference fluidity or prefactor, $Q$ the activation energy, $R$ the gas constant, and $T$ the temperature (units of K). Following Cuffey and Paterson (2010), we set the reference fluidity to $A_0 = 1.258 \times 10^{13}\,\mathrm{MPa}^{-3}\,\mathrm{a}^{-1}$ and the activation energy to $Q = 60\,\mathrm{kJ\,mol}^{-1}$ for $T < 263\,(\mathrm{K} - 10°\mathrm{C})$, and $A_0 = 6.046 \times 10^{28}\,\mathrm{MPa}^{-3}\,\mathrm{a}^{-1}$ and $Q = 139\,\mathrm{kJ\,mol}^{-1}$ for $T > 263\,\mathrm{K}$. The code is based on a 3D Finite Element Method (FEM) for numerically solving the Stokes equations, computing ice flow by solving Eq. (B1) subject to the principle of mass conservation, $\boldsymbol{\nabla} \cdot \boldsymbol{u} = 0$.

In the context of continuum mechanics, fractures and crevasses that weaken the ice can be represented by reducing the enhancement factor $E_D$ following:

$$E_D = \frac{1}{(1-D)^n} \tag{B6}$$

with $D \in [0, 1[$ (see Sect. B3). In our application of Sect. 4, we keep $E_D = 1$, i.e. we do not account for the retroaction of the evolving damage on the $E_D$.

For transient simulations, the advection equation of the surface can be solved (Gagliardini et al., 2013). We apply a Dirichlet boundary condition on the velocity at the inflow boundary:

$$\boldsymbol{u} \cdot \boldsymbol{n} = 0, \tag{B7}$$

where $\boldsymbol{n}$ is the normal to the surface. At the ocean-ice interface, we apply a sea pressure:

$$p_{\mathrm{w}} = \rho_w g z(t) \tag{B8}$$

where $\rho_w$ is the seawater density and $z(t)$ is the depth, resulting in the following Neumann condition applied on the ice-ocean interface:

$$\boldsymbol{\sigma} \cdot \boldsymbol{n} = -p_{\mathrm{w}} \boldsymbol{n}. \tag{B9}$$

### B2 Ice flow model initialization

The 3D model is initialized by calculating the fluidity of the ice as a function of the temperature following an Arrhenius' law (e.g. Gillet-Chaulet et al., 2012; Mosbeux et al., 2020) and by reconstructing a poorly known parameter in ice sheet models, the basal drag at the interface between the ice and the bedrock (that can vary depending on the basal condition and the presence of water at the interface). The reconstruction consists in finding the parameter values that minimize the discrepancy between observed and model surface velocities (for a given geometry), following the approach presented in Macayeal (1993) and commonly used in ice sheet modeling and in Elmer/Ice in particular (e.g. Mosbeux et al., 2023, 2016; Hill et al., 2023; Klein et al., 2020; Brondex et al., 2019). This inversion often leads to unrealistic ice flux divergence (Seroussi et al., 2011), caused by remaining uncertainties in model initial conditions, that we mitigate by running the model forward over 20 years, damping the divergences to acceptable values (e.g. Gillet-Chaulet et al., 2012).

The 3D finite element mesh is built in two steps. First, we build a 2D-plan mesh of the footprint of the glacier basin preferentially refined along directions of highest second-derivative of observed ice velocity and ice thickness with a resolution varying from 1 to 25 km (e.g. Hill et al., 2023). The 2D mesh is then vertically extruded into 11 layers, with the bottom and top surfaces adjusted to match the glacier bed and surface digital elevation models, leading to a vertical resolution ranging from 10 to 300 m locally depending on the ice thickness (e.g. Gillet-Chaulet et al., 2012). This results in a 3D mesh of 651 690 linear wedge elements and 316 310 nodes. Due to the large number of elements and nodes, the simulations are conducted in parallel with 32 partitions, giving roughly 10 000 nodes per partition.

### B3 Damage Mechanics

The increase of damage in the media depends on the stress field and occurs when the maximum tensile stress exceeds a threshold $\sigma_{\mathrm{th}}$ (between 0.01–0.20 MPa in Krug et al., 2014). To account for ice heterogeneity, some noise can be introduced on $\sigma_{\mathrm{th}}$: $\sigma_{\mathrm{th}} = (\bar{\sigma}_{\mathrm{th}}) + \delta\sigma$, where $\delta\sigma/(\bar{\sigma}_{\mathrm{th}})$ follows a standard normal distribution with an arbitrary standard deviation. For a better assessment of the impact of the SL time stepping choice, we keep $\delta\sigma = 0$ in all our simulations. As stated in Sect. 4, all our simulations are conducted with $\sigma_{\mathrm{th}} = 0.65$ MPa for the purpose of the experiment. However, for an accurate representation of damage and a better alignment with current observations, this threshold should be carefully considered in relation to the tensile strength of ice and the selected damage criterion (Mercenier et al., 2018, 2019).

The source term can be described as follows:

$$f(\chi) = \begin{cases} B \cdot \chi(\sigma_I, \sigma_{\mathrm{th}}, D) & \text{if } \chi \geq 0 \\ 0 & \text{if } \chi < 0 \end{cases}, \tag{B10}$$

where $B$ is a damage enhancement factor that needs to be calibrated and that we take equal to 1 in our experiment. The variable $\chi$ is called the damage criterion and writes:

$$\chi(\sigma_I, \sigma_{\mathrm{th}}, D) = \frac{\sigma_I}{1-D} - \sigma_{\mathrm{th}} + p_{\mathrm{w}}, \tag{B11}$$

where $\sigma_I$ is the maximal tensile principal stress, and $p_{\mathrm{w}}$ is the eventual water pressure in the crevasse (e.g. sea pressure in basal crevasses). In our application of Sect. 4, we keep $p_{\mathrm{w}} = 0$, hence preventing damage creation close to the bed. We can see $\chi$ increases as $D$ increases, simulating the fact that internal forces acting on any damaged section of material are the same as the ones before damage but on a reduced surface (Lemaitre and Chaboche, 1978).

*Code and data availability.* The Last released version of the Elmer/Ice code is publicly available on Zenodo at (v26.2.1, https://doi.org/10.5281/zenodo.19888172, Ruokolainen et al., 2026). All simulations were performed with Elmer/Ice based on commit 97773a1f2 with some additional chances that have been included since v26.1. We encourage users to use the latest version available on the Elmer/Ice Github repository and refer to the documentation for recent updates. All the material necessary to reproduce the simulations is available through CM GitHub (https://github.com/cmosbeux/SEMI-LAGRANGIAN-PUBLICATION.git; Mosbeux, 2025b) and on Zenodo at https://doi.org/10.5281/zenodo.15741827 (Mosbeux, 2025a), along with post-processing python scripts and detailed explanations.

*Video supplement.* A video supplement for the ice damage application (Sect. 4) is available in the video section of the Zenodo repository: https://doi.org/10.5281/zenodo.15741827 (Mosbeux, 2025a).

*Author contributions.* PR developed the initial version of the semi-Lagrangian solver in Elmer, CM and JB helped in resolving bugs and implementing solutions. CM, JB, AG, and FGC developed the different test cases. CM conducted the simulations and the analysis with help from all the authors. All authors contributed to the writing of the manuscript.

*Competing interests.* The contact author has declared that none of the authors has any competing interests.

*Acknowledgements.* The authors thank the editor, Ludovic Räss, as well as Ravindra Duddu and two anonymous reviewers for their insightful and helpful comments. These projects received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no. 820575 (TiPACCs) and no. 869304 (PROTECT contribution 171) respectively. This study has received funding from Agence Nationale de la Recherche – France 2030 as part of the PEPR TRACCS programme under grant number ANR-22-EXTR-0010. The Elmer/Ice computations presented in this paper were performed using the HPC resources of TGCC under the allocations A0140106035, AD010106066R1 and AD010106066R2 attributed by GENCI. We also acknowledge the French National Research Infrastructure CLIMERI-France (https://climeri-france.fr/, last access: 15 June 2026) which supports the French contribution to the CMIP and CORDEX international modeling exercises, and provides national label for the code Elmer/Ice.

*Financial support.* This research has been supported by the EU Horizon 2020 (grant nos. 820575 and 869304) and the Agence Nationale de la Recherche (ISClim (grant no. ANR-22-EXTR-0010)).

*Review statement.* This paper was edited by Ludovic Räss and reviewed by Albert de Montserrat Navarro, Ravindra Duddu, and one anonymous referee.

## References

Albrecht, T. and Levermann, A.: Fracture field for large-scale ice dynamics, J. Glaciol., 58, 165–176, https://doi.org/10.3189/2012JoG11J191, 2012.

Alley, K. E., Scambos, T. A., Alley, R. B., and Holschuh, N.: Troughs developed in ice-stream shear margins precondition ice shelves for ocean-driven breakup, Science Advances, 5, eaax2215, https://doi.org/10.1126/sciadv.aax2215, 2019.

Berg, B. and Bassis, J.: Crevasse advection increases glacier calving, J. Glaciol., 68, 977–986, https://doi.org/10.1017/jog.2022.10, 2022.

Brezzi, F., Marini, L. D., and Süli, E.: Discontinuous galerkin methods for first-order hyperbolic problems, Math. Mod. Meth. Appl. S., 14, 1893–1903, https://doi.org/10.1142/S0218202504003866, 2004.

Brondex, J., Gillet-Chaulet, F., and Gagliardini, O.: Sensitivity of centennial mass loss projections of the Amundsen basin to the friction law, The Cryosphere, 13, 177–195, https://doi.org/10.5194/tc-13-177-2019, 2019.

Brus, S. R., Wirasaet, D., Westerink, J. J., and Dawson, C.: Performance and scalability improvements for discontinuous Galerkin solutions to conservation laws on unstructured grids, J. Sci. Comput., 70, 210–242, https://doi.org/10.1007/s10915-016-0249-y, 2017.

Cheng, G., Morlighem, M., and Gudmundsson, G. H.: Numerical stabilization methods for level-set-based ice front migration, Geosci. Model Dev., 17, 6227–6247, https://doi.org/10.5194/gmd-17-6227-2024, 2024.

Cuffey, K. M. and Paterson, W. S. B.: The Physics of Glaciers, Academic Press, New York, ISBN 978-0-12-369461-4, 2010.

Côté, J. and Staniforth, A.: A two-time-level semi-Lagrangian semi-implicit scheme for spectral models, Mon. Weather Rev., 116, 2003–2012, https://doi.org/10.1175/1520-0493(1988)116<2003:ATTLSL>2.0.CO;2, 1988.

Dapogny, C., Dobrzynski, C., and Frey, P.: Three-dimensional adaptive domain remeshing, implicit domain meshing, and applications to free and moving boundary problems, J. Comput. Phys., 262, 358–378, https://doi.org/10.1016/j.jcp.2014.01.005, 2014.

Gagliardini, O. and Meyssonnier, J.: Flow simulation of a firn-covered cold glacier, Ann. Glaciol., 24, 242–248, https://doi.org/10.3189/S0260305500012246, 1997.

Gagliardini, O., Zwinger, T., Gillet-Chaulet, F., Durand, G., Favier, L., de Fleurian, B., Greve, R., Malinen, M., Martín, C., Råback, P., Ruokolainen, J., Sacchettini, M., Schäfer, M., Seddik, H., and Thies, J.: Capabilities and performance of Elmer/Ice, a new-generation ice sheet model, Geosci. Model Dev., 6, 1299–1318, https://doi.org/10.5194/gmd-6-1299-2013, 2013.

Gilbert, A., Gagliardini, O., Vincent, C., and Wagnon, P.: A 3-D thermal regime model suitable for cold accumulation zones of polythermal mountain glaciers, J. Geophys. Res.-Earth, 119, 1876–1893, https://doi.org/10.1002/2014JF003199, 2014.

Gilbert, A., Vincent, C., Gagliardini, O., Krug, J., and Berthier, E.: Assessment of thermal change in cold avalanching glaciers in relation to climate warming, Geophys. Res. Lett., 42, 6382–6390, https://doi.org/10.1002/2015GL064838, 2015.

Gillet-Chaulet, F., Gagliardini, O., Meyssonnier, J., Zwinger, T., and Ruokolainen, J.: Flow-induced anisotropy in polar ice and related ice-sheet flow modelling, J. Non-Newton. Fluid, 134, 33–43, https://doi.org/10.1016/j.jnnfm.2005.11.005, 2006.

Gillet-Chaulet, F., Hindmarsh, R. C. A., Corr, H. F. J., King, E. C., and Jenkins, A.: In-situquantification of ice rheology and direct measurement of the Raymond Effect at Summit, Greenland using a phase-sensitive radar, Geophys. Res. Lett., 38, L24503, https://doi.org/10.1029/2011GL049843, 2011.

Gillet-Chaulet, F., Gagliardini, O., Seddik, H., Nodet, M., Durand, G., Ritz, C., Zwinger, T., Greve, R., and Vaughan, D. G.: Greenland ice sheet contribution to sea-level rise from a new-generation ice-sheet model, The Cryosphere, 6, 1561–1576, https://doi.org/10.5194/tc-6-1561-2012, 2012.

Glen, J. W.: The creep of polycrystalline ice, P. Roy. Soc. A-Math. Phy., 228, 519–538, https://doi.org/10.1098/rspa.1955.0066, 1955.

Grinsted, A., Rathmann, N. M., Mottram, R., Solgaard, A. M., Mathiesen, J., and Hvidberg, C. S.: Failure strength of glacier ice inferred from Greenland crevasses, The Cryosphere, 18, 1947–1957, https://doi.org/10.5194/tc-18-1947-2024, 2024.

Haselbacher, A., Najjar, F. M., and Ferry, J. P.: An efficient and robust particle-localization algorithm for unstructured grids, J. Comput. Phys., 225, 2198–2213, https://doi.org/10.1016/j.jcp.2007.03.018, 2007.

Hill, E. A., Urruty, B., Reese, R., Garbe, J., Gagliardini, O., Durand, G., Gillet-Chaulet, F., Gudmundsson, G. H., Winkelmann, R., Chekki, M., Chandler, D., and Langebroek, P. M.: The stability of present-day Antarctic grounding lines – Part 1: No indication of marine ice sheet instability in the current geometry, The Cryosphere, 17, 3739–3759, https://doi.org/10.5194/tc-17-3739-2023, 2023.

Hughes, T. J. R.: Recent progress in the development and understanding of SUPG methods with special reference to the compressible Euler and Navier-Stokes equations, Int. J. Numer. Meth. Fl., 7, 1261–1275, https://doi.org/10.1002/fld.1650071108, 1987.

Huth, A., Duddu, R., and Smith, B.: A generalized interpolation material point method for shallow ice shelves. 1: Shallow shelf approximation and ice thickness evolution, J. Adv. Model. Earth Sy., 13, e2020MS002277, https://doi.org/10.1029/2020MS002277, 2021a.

Huth, A., Duddu, R., and Smith, B.: A generalized interpolation material point method for shallow ice shelves. 2: Anisotropic nonlocal damage mechanics and rift propagation, J. Adv. Model. Earth Sy., 13, e2020MS002292, https://doi.org/10.1029/2020MS002292, 2021b.

Huth, A., Duddu, R., Smith, B., and Sergienko, O.: Simulating the processes controlling ice-shelf rift paths using damage mechanics, J. Glaciol., 69, 1915–1928, https://doi.org/10.1017/jog.2023.71, 2023.

Jiménez, S., Duddu, R., and Bassis, J.: An updated-Lagrangian damage mechanics formulation for modeling the creeping flow and fracture of ice sheets, Comput. Method. Appl. M., 313, 406–432, https://doi.org/10.1016/j.cma.2016.09.034, 2017.

Jouvet, G., Röllin, S., Sahli, H., Corcho, J., Gnägi, L., Compagno, L., Sidler, D., Schwikowski, M., Bauder, A., and Funk, M.: Mapping the age of ice of Gauligletscher combining surface radionuclide contamination and ice flow modeling, The Cryosphere, 14, 4233–4251, https://doi.org/10.5194/tc-14-4233-2020, 2020.

Ketefian, G. S., Gross, E. S., and Stelling, G. S.: Accurate and consistent particle tracking on unstructured grids, Int. J. Numer. Meth. Fl., 80, 648–665, https://doi.org/10.1002/fld.4168, 2016.

Klein, E., Mosbeux, C., Bromirski, P. D., Padman, L., Bock, Y., Springer, S. R., and Fricker, H. A.: Annual cycle in flow of Ross Ice Shelf, Antarctica: contribution of variable basal melting, J. Glaciol., 66, 861–875, https://doi.org/10.1017/jog.2020.61, 2020.

Krug, J., Weiss, J., Gagliardini, O., and Durand, G.: Combining damage and fracture mechanics to model calving, The Cryosphere, 8, 2101–2117, https://doi.org/10.5194/tc-8-2101-2014, 2014.

Kuzmin, D.: On the design of general-purpose flux limiters for finite element schemes. I. Scalar convection, J. Comput. Phys., 219, 513–531, https://doi.org/10.1016/j.jcp.2006.03.034, 2006.

Kuzmin, D.: A vertex-based hierarchical slope limiter for $p$-adaptive discontinuous Galerkin methods, J. Comput. Appl. Math., 233, 3077–3085, https://doi.org/10.1016/j.cam.2009.05.028, 2010.

Lemaitre, J. and Lemaitre, J. and Chaboche, J. L. Aspect phénoménologique de la rupture par endommagement, Journal de Mécanique Appliquée, 2, 317–365, 1978.

Lhermitte, S., Sun, S., Shuman, C., Wouters, B., Pattyn, F., Wuite, J., Berthier, E., and Nagler, T.: Damage accelerates ice shelf instability and mass loss in Amundsen Sea Embayment, P. Natl. Acad. Sci. USA, 117, 24735–24741, https://doi.org/10.1073/pnas.1912890117, 2020.

Li, Y., Coulon, V., Blasco, J., Qiao, G., Yang, Q., and Pattyn, F.: Damage intensity increases ice mass loss from Thwaites Glacier, Antarctica, The Cryosphere, 19, 4373–4390, https://doi.org/10.5194/tc-19-4373-2025, 2025.

Löfgren, A., Ahlkrona, J., and Helanow, C.: Increasing stable time-step sizes of the free-surface problem arising in ice-sheet simulations, Journal of Computational Physics: X, 16, 100114, https://doi.org/10.1016/j.jcpx.2022.100114, 2022.

Macayeal, D. R.: A tutorial on the use of control methods in ice-sheet modeling, J. Glaciol., 39, 91–98, https://doi.org/10.3189/S0022143000015744, 1993.

Macpherson, G. B., Nordin, N., and Weller, H. G.: Particle tracking in unstructured, arbitrary polyhedral meshes for use in CFD and molecular dynamics, Commun. Numer. Meth. En., 25, 263–273, https://doi.org/10.1002/cnm.1128, 2009.

Masud, A. and Khurram, R. A.: A multiscale/stabilized finite element method for the advection–diffusion equation, Comput. Method. Appl. M., 193, 1997–2018, https://doi.org/10.1016/j.cma.2003.12.047, 2004.

Mercenier, R., Lüthi, M. P., and Vieli, A.: Calving relation for tidewater glaciers based on detailed stress field analysis, The Cryosphere, 12, 721–739, https://doi.org/10.5194/tc-12-721-2018, 2018.

Mercenier, R., Lüthi, M. P., and Vieli, A.: A transient coupled ice flow-damage model to simulate iceberg calving from tidewater outlet glaciers, J. Adv. Model. Earth Sy., 11, 3057–3072, https://doi.org/10.1029/2018MS001567, 2019.

Mortezazadeh, M., Cossette, J.-F., Dastoor, A., de Grandpré, J., Ivanova, I., and Qaddouri, A.: Sweep interpolation: a cost-effective semi-Lagrangian scheme in the Global Environmental Multiscale model, Geosci. Model Dev., 17, 335–346, https://doi.org/10.5194/gmd-17-335-2024, 2024.

Mosbeux, C.: Semi-Lagrangian Advection Scheme in Elmer/Ice – Benchmark and Damage Tests, Zenodo [code, data set, video], https://doi.org/10.5281/zenodo.15741827, 2025a.

Mosbeux, C.: Semi-Lagrangian Advection Scheme in Elmer/Ice – Benchmark and Damage Tests, GitHub [data set], https://github.com/cmosbeux/SEMI-LAGRANGIAN-PUBLICATION (last access: 15 June 2026), 2025b.

Mosbeux, C., Gillet-Chaulet, F., and Gagliardini, O.: Comparison of adjoint and nudging methods to initialise ice sheet model basal conditions, Geosci. Model Dev., 9, 2549–2562, https://doi.org/10.5194/gmd-9-2549-2016, 2016.

Mosbeux, C., Wagner, T. J. W., Becker, M. K., and Fricker, H. A.: Viscous and elastic buoyancy stresses as drivers of ice-shelf calving, J. Glaciol., 66, 643–657, https://doi.org/10.1017/jog.2020.35, 2020.

Mosbeux, C., Padman, L., Klein, E., Bromirski, P. D., and Fricker, H. A.: Seasonal variability in Antarctic ice shelf velocities forced by sea surface height variations, The Cryosphere, 17, 2585–2606, https://doi.org/10.5194/tc-17-2585-2023, 2023.

Ouardghi, A., El-Amrani, M., and Seaid, M.: An adaptive enriched semi-Lagrangian finite element method for coupled flow-transport problems, Comput. Fluids, 240, 105474, https://doi.org/10.1016/j.compfluid.2022.105474, 2022.

Priestley, A.: A quasi-conservative version of the semi-Lagrangian advection scheme, Mon. Weather Rev., 121, 621–629, https://doi.org/10.1175/1520-0493(1993)121<0621:AQCVOT>2.0.CO;2, 1993.

Ranganathan, M., Robel, A. A., Huth, A., and Duddu, R.: Glacier damage evolution over ice flow timescales, The Cryosphere, 19, 1599–1619, https://doi.org/10.5194/tc-19-1599-2025, 2025.

Reed, W. H. andHill, T. R.: Triangular Mesh Methods for the Neutron Transport Equation, Los Alamos Scientific Laboratory Report LA-UR-73-479, OSTI 4491151, presented at the American Nuclear Society Topical Meeting on Mathematical Models and Computational Techniques for Analysis of Nuclear Systems, Ann Arbor, Michigan, 1973.

Ruokolainen, J., Råback, P., Malinen, M., Zwinger, T., Kataja, J., Ilvonen, S., Lyly, M., Byckling, M., Takala, E., Gillet-Chaulet, F., Gagliardini, O., Todd, J., Gladstone, R., Gong, C., Cook, S., Robertsen, F., Wheel, I., Chekki, M., Ponomarev, P., van Dongen, E., Thies, J., Saeki, T., Löfgren, A., Rodenberg, B., and Schannwell, C.: ElmerFEM, Zenodo [code], https://doi.org/10.5281/zenodo.19888172, 2026.

Samelson, R. M. and Wiggins, S.: Lagrangian Transport in Geophysical Jets and Waves: The Dynamical Systems Approach, in: Interdisciplinary Applied Mathematics, Vol. 31, Springer, New York, https://doi.org/10.1007/978-0-387-46213-4, 2006.

Seroussi, H., Morlighem, M., Rignot, E., Larour, E., Aubry, D., Ben Dhia, H., and Kristensen, S. S.: Ice flux divergence anomalies on 79north Glacier, Greenland, Geophys. Res. Lett., 38, L09501, https://doi.org/10.1029/2011GL047338, 2011.

Smith, B., Fricker, H. A., Gardner, A. S., Medley, B., Nilsson, J., Paolo, F. S., Holschuh, N., Adusumilli, S., Brunt, K., Csatho, B., Harbeck, K., Markus, T., Neumann, T., Siegfried, M. R., and Zwally, H. J.: Pervasive ice sheet mass loss reflects competing ocean and atmosphere processes, Science, 368, 1239–1242, https://doi.org/10.1126/science.aaz5845, 2020.

Sun, S. and Gudmundsson, G. H.: The speedup of Pine Island Ice Shelf between 2017 and 2020: revaluating the importance of ice damage, J. Glaciol., 69, 1983–1991, https://doi.org/10.1017/jog.2023.76, 2023.

Sun, S., Cornford, S. L., Moore, J. C., Gladstone, R., and Zhao, L.: Ice shelf fracture parameterization in an ice sheet model, The Cryosphere, 11, 2543–2554, https://doi.org/10.5194/tc-11-2543-2017, 2017.

Van Liefferinge, B. and Pattyn, F.: Using ice-flow models to evaluate potential sites of million year-old ice in Antarctica, Clim. Past, 9, 2335–2345, https://doi.org/10.5194/cp-9-2335-2013, 2013.

Weertman, J.: Creep deformation of ice, Annu. Rev. Earth Pl. Sc., 11, 215–240, https://doi.org/10.1146/annurev.ea.11.050183.001243, 1983.

Wirbel, A., Jarosch, A. H., and Nicholson, L.: Modelling debris transport within glaciers by advection in a full-Stokes ice flow model, The Cryosphere, 12, 189–204, https://doi.org/10.5194/tc-12-189-2018, 2018.

Zalesak, S. T.: Fully multidimensional flux-corrected transport algorithms for fluids, J. Comput. Phys., 31, 335–362, https://doi.org/10.1016/0021-9991(79)90051-2, 1979.

Zienkiewicz, O. C., Taylor, R. L., Sherwin, S. J., and Peiró, J.: On discontinuous Galerkin methods, Int. J. Numer. Meth. Eng., 58, 1119–1148, https://doi.org/10.1002/nme.884, 2003.