Eurographics Symposium on Geometry Processing 2018
T. Ju and A. Vaxman
(Guest Editors)

Volume 37 (2018), Number 5

An explicit structure-preserving numerical scheme for EPDiff

Omri Azencot[1] Orestis Vantzos[2] Mirela Ben-Chen[2]

[1] University of California, Los Angeles
[2] Technion – Israel Institute of Technology

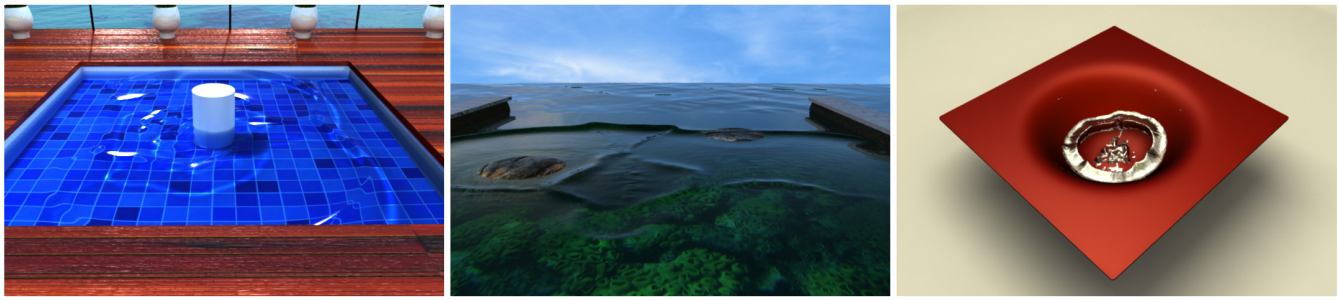

Figure 1: *Results obtained with our method. See the corresponding figures (Figs. 15,12, and 10) for additional details.*

Abstract

We present a new structure-preserving numerical scheme for solving the Euler–Poincaré Differential (EPDiff) equation on arbitrary triangle meshes. Unlike existing techniques, our method solves the difficult non-linear EPDiff equation by constructing energy preserving, yet fully explicit*, update rules. Our approach uses standard differential operators on triangle meshes, allowing for a simple and efficient implementation. Key to the structure-preserving features that our method exhibits is a novel numerical splitting scheme. Namely, we break the integration into three steps which rely on linear solves with a fixed sparse matrix that is independent of the simulation and thus can be pre-factored. We test our method in the context of simulating concentrated reconnecting wavefronts on flat and curved domains. In particular, EPDiff is known to generate geometrical fronts which exhibit wave-like behavior when they interact with each other. In addition, we also show that at a small additional cost, we can produce globally-supported periodic waves by using our simulated fronts with wavefronts tracking techniques. We provide quantitative graphs showing that our method exactly preserves the energy in practice. In addition, we demonstrate various interesting results including annihilation and recreation of a circular front, a wave splitting and merging when hitting an obstacle and two separate fronts propagating and bending due to the curvature of the domain.*

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Physically based modeling

1. Introduction

Structure-preserving integrators [HLW06] play an important role in many applications in pure and applied mathematics. These methods are unique in that they solve a differential equation while keeping certain geometrical invariant properties of the system. Numerical integrators that respect the underlying invariants of the dynamics are especially important in computer graphics, where gain/loss of vorticity [AWO*14] or kinetic energy [AVW*15, MCP*09] may lead to inaccurate long-integration and undesirable visual artifacts. The goal of this paper is to suggest an efficient structure-preserving numerical scheme for solving the challenging EPDiff equation.

EPDiff arises in several applications and domains. For example, in computational anatomy, EPDiff are the governing equations in the diffeomorphic deformation framework [MTY02], where the goal is to compute a non-linear deformation between a given pair of source and target images. In fluid mechanics, EPDiff is used in the context of turbulence with the so-called Navier–Stokes-α model. A similar formulation was devised to better handle an effect known as energy cascading [LMH*15]. Perhaps closest to our focus is the link to shallow water wave dynamics, where EPDiff are used to model ocean wavefronts, 100–200 km in length (see the images in [HS13]). We also use EPDiff to model interacting surface waves.

Intuitively, EPDiff describes the evolution of "waves" on a fixed domain that can reconnect after collision, i.e., concentrated waves can pass through each other and maintain their shape. To capture the reconnection effect, EPDiff encodes the following key idea. Upon collision, an exchange of *momentum* occurs between the involved fronts. Figure 2 illustrates this behavior for 1D singular waves (*peakons*) where the initial peaked solitons "switch" places due to exchange of momentum, and we further show a corresponding 2D example in Figure 11. The derived model can be interpreted as the advection of the concentrated momentum over the velocity field (a *non-linear* term) where the velocity is a smoothened version of the momentum (a *non-local* term). Interestingly, the above setup is reminiscent of the vorticity equation (see e.g., [Saf92]) which governs the kinematics of ideal incompressible flows where vorticity is being transported by the velocity and these quantities are linked through the Biot–Savart law. We provide a further elaborate discussion on the similarities between the models in Section 2.

The Euler–Poincaré (EPDiff) equation has received increasing attention in the literature (see e.g., [HS13]). However, while there are some works which manage to discretize this complex PDE in various configurations, its discretization is still considered a non-trivial task since it involves several challenges. For example, any discrete method must be relatively accurate because the advected momentum is highly concentrated and thus discretization errors become visible quite quickly. Moreover, the stability and behaviour of the fronts depend heavily on the particular non-linearity of the equation; if that is not discretized correctly, the concentrated waves will be unstable regardless of how numerically accurate the discretization is. Similarly, time integration is equally important as it is expected to preserve the properties of the continuous problem as much as possible. Finally, the spatial differential operators should gracefully handle boundaries and deal with curved domains.

To better assess the proposed approach, we will try to classify it with respect to other methods for simulation of fronts and waves. From a broader perspective, our model can be considered as part of the family of *shallow water equations* (see e.g., [Vre13]). In these models, the assumptions of columnar motion and averaged velocity over the fluid height naturally lead to a reduction of dimensionality. Thus, although the 3D Navier–Stokes (NS) equations could capture the effects we are interested in, the involved computational cost is too prohibitive for practical uses when compared to a 2D model such as ours [Bri08]. Moreover, qualitative properties are usually hard to infer from the general NS model. For instance, certain singular solutions are known to exist for our model, allowing for a better qualitative and quantitative understanding of the model.

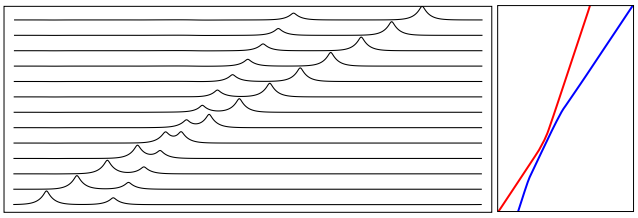

Figure 2: Momentum exchange. *The 1D simulation illustrates how two peakons with different momenta interact over space (horizontal axis) and time (vertical axis), left. Notice that the left peakon transfers momentum to the right peakon when they collide. Paths of the left/right soliton peaks are shown with red/blue curves on the right.*

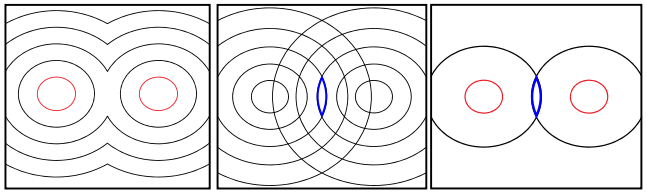

Figure 3: *Tracking fronts shown here as red curves can be done using the eikonal eq. leading to viscosity solutions (left). Alternatively, periodic functions yield superimposed interaction between waves denoted by the blue curve (middle). Our method tracks concentrated waves which interact with other waves (right). See also Fig. 5.*

Alternative approaches to wave simulation are commonly based on insights from linearized water wave theory (see e.g., [DD91]). At the core of the linearized model we are given descriptions of the wave *function* as a sum of sinusoidal functions and of the wave *propagation speed* as the solution of the eikonal equation. Numerical methods based on this framework can be categorized into the following two groups. The first group of techniques treat the propagated front as a geometrical wave and thus obtain viscosity solutions, i.e., concentrated fronts are possible, however, the superposition principle of waves does not hold, see Fig. 3, left. Schemes from the second group approximate the wave function, and the obtained results do exhibit superposition, but modeling concentrated waves is difficult, Fig. 3, middle. Our method enjoys the advantages of both approaches, thus achieving superimposed concentrated waves as we illustrate in Fig. 3, right.

In this work, we present a fully (time- and space-) discrete scheme for the EPDiff equation, with the explicit aim that it is *structure-preserving*, i.e., that it conserves a physically appropriate energy, and that it is applicable to general meshes of complex surfaces. The first goal is achieved via a novel time integrator, which despite being fully *explicit*, is energy preserving (and therefore stable) by construction, under some mild conditions. Furthermore, the scheme is based on standard discrete (differential and interpolation) operators that are well-behaved on unstructured triangulations of curved surfaces. Overall, our method is efficient as it involves, per step, at most three sparse linear solves with a *fixed* matrix, that depends only on the mesh and thus can be pre-factored for the entire simulation. Moreover, as a post-processing step, we can couple the wavefronts tracked by our scheme with periodic waves of various speeds, to achieve a combined reconnection-superposition-dispersal effect. Finally, we show several results including bending of waves due to curvature effects, plausible behavior of waves interacting with boundaries, and reconnection of concentrated fronts after collision.

1.1. Related Work

Mathematical mechanics. The EPDiff equation has received increasing attention over the last decade, and the interested reader is referred to the book [HSSE09] which presents theoretical aspects of the problem, as well as several applications. Unfortunately, there exist only a few numerical methods which discretize this equation, possibly due to its complexity and related challenges. For instance, it is known [HM05] that solutions might develop discontinuities, even though the initial conditions are smooth (we also show it in, e.g., Fig. 10). Nevertheless, various particle methods approximate the solution as a linear combination of Dirac functions and exhibit momentum preservation [CTM12] or apply different boundary conditions [CKL14]. The Eulerian approach presented in [HS13] uses an explicit Runge–Kutta temporal integrator on logically rectangular flat domains, and thus, it is not structure-preserving. Probably closest to our method is the technique which was recently suggested by Larsson et al. [LMMM16]. They introduce structure-preserving schemes for EPDiff on a regular structured grid. Their method preserves energy but it is not time invertible. In addition, extending their method to support curvature effects is non-trivial, as it would require generalizing it to handle triangle meshes which are frequently used for representing curved domains.

Fluid simulation. The topic of ocean and wave simulation is well researched in computer graphics and reviewing the vast associated literature is beyond the scope of this paper. In our discussion, we mainly focus on methods which specifically model waves and their interaction, and we refer to the available reviews [Bri08, DCGG11] which provide an extensive discussion. Moreover, we emphasize that we only suggest to capture wave-like phenomena using EPDiff as a testing environment for our numerical scheme. In practice, while our approach may provide certain advantages over existing techniques, it should be generally considered as a complementary approach. We base our summary of related-work on the following classification. In the linear model of waves, the function whose gradient determines the propagation speed is known as the *phase function*. Mathematically, describing the aforementioned reconnection effect is equivalent to requiring a *multi-valued* phase function, whereas *single-valued* phase functions discard wave interactions, i.e., wave reconnection is lost. In what follows, we divide methods depending on whether their approximated phase function is multi-valued or single-valued.

A popular approach for phase function approximation is to assume it is a sum of fixed propagation velocities [MWM*87], which can be further improved [T*01, HNC02]. A natural extension of the former approach aims for a better integration of the eikonal equation by assuming varying speeds [FR86, Pea86], and consequently, it allows for additional wave effects. Another common alternative is to employ reductions such as shallow water equations [KM90] whose extensions enable intricate effects of interaction with rigid and soft bodies [CM10] and even breaking of waves [TMFSG07]. Keeler and Bridson [KB14] simulate deep ocean waves by solving for a potential flow using a boundary integral equation. Yuksel et al. [YHK07] present an interesting approach where fronts are represented using particles achieving interaction with dynamic objects at realtime rates. Other works attempt to design waves using guide shapes [NB11] or through fitting a desired look [NSB13].

Supporting multi-valued phase functions can be achieved through solving the volumetric Navier–Stokes eqs. [FF01]. However, while this method captures several effects and in particular concentrated fronts are doable, the involved computational cost is too restrictive when compared to other methods. *iWave* [Tes04] offers an attractive framework for achieving wave behaviors as diffraction and refraction. Recently, Jeschke and Wojtan [JW15] facilitated a Lagrangian technique known as *wavefront tracking* to reconstruct high-frequency wave functions, and to obtain interesting wave behaviors while improving on former methods [GLS00, TB87, GM02].

Computational imaging. Finally, we briefly mention that the EPDiff equation also appears in computational anatomy in shape matching problems. Given initial and final outlines of an image, the goal is to interpolate between the input outlines using an optimization problem whose solution satisfies the EPDiff equation. An attractive application of this framework is the 3D reconstruction of the human brain from 2D PET scans [BMTY05]. However, since the above setup involves boundary value problems, it requires other approaches than those used for initial value problems as ours, see e.g., [AVBC16]. Therefore, we only point out this interesting link and refer to a detailed review on the subject [MTY02].

1.2. Contributions

The contributions of our method can be summarized as follows.

Discrete EPDiff on curved domains. Our method is based on an efficient discretization of the EPDiff PDE by devising a novel fully explicit splitting scheme for the temporal integration which is energy preserving by construction. Additionally, our method offers a tradeoff between exactly maintaining the velocity-momentum non-local relation or obtaining a time invertible scheme. Finally, our spatial discretization makes use of standard operators defined directly on triangle meshes and thus can be easily implemented.

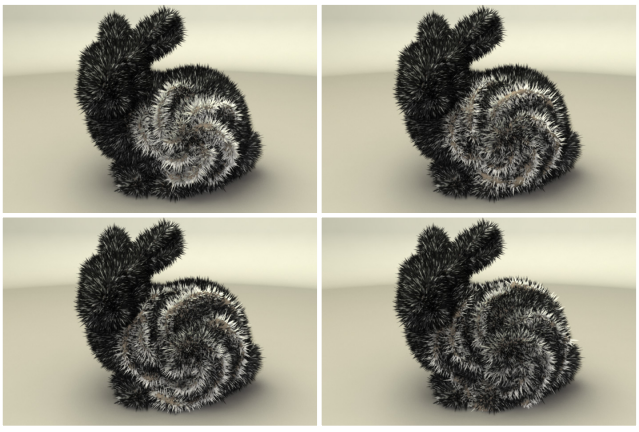

Figure 4: *Our method is adapted to work on curved geometries, allowing to propagate general initial wave profiles.*

Concentrated reconnecting wavefronts. We simulate geometrical fronts whose behavior closely reflects the non-linear interaction between waves. Our fronts are different from the viscosity solutions arising from the eikonal equation. In addition, we convolve the simulated data with various wave sources to support globally-supported periodic waves in a fashion similar to wavefront tracking methods.

Non-trivial effects. We show several intricate results on flat and curved domains including annihilation and recreation of a circular front (Fig. 10), a wave splitting to two as it interacts with an obstacle (Fig. 15), waves exhibiting different spatial profiles due to the underlying curvature (Fig. 12) and other interesting results.

2. Smooth Setting

We seek to simulate, via the EPDiff equation, the *motion of wavefronts* such that the resulting fronts exhibit profiles which are realistic, see e.g., Fig. 5. Although the equation itself is the product of a rather complicated chain of reductions, regularizations and approximations of the Navier–Stokes equations, we will try to motivate it here in a more direct manner.

The key intuition is that an expanding wavefront can be seen as a self-propagating concentration of linear momentum, much in the same way that *a vortex is a self-propagating concentration of vorticity* (i.e., angular momentum). In other words, a wavefront forces the fluid to move linearly, just as the vortex forces the fluid to rotate around it. An initial set of vortices, each localized along a curve (vortex filament) or a hypersurface (vortex sheet), interact in a non-local manner as they are advected by the sum of their individual induced velocity fields. In the case of an inviscid and incompressible fluid (the Euler equation regime), this motion is described by the *vorticity equation* (see e.g., [ZBG15]), which can be written as

$$\underbrace{\frac{\partial \omega}{\partial t} + v \cdot \nabla \omega}_{\text{advection}} - \underbrace{\omega \cdot \nabla v}_{\text{stretching}} = 0 \,. \tag{1}$$

This PDE is used to update the vorticity, given a velocity field, via a combination of transporting and stretching. To close the system we need to specify the relation between the velocity and the vorticity, which is given by $\omega = \nabla \times v$. In integral form, this relation is the *Biot–Savart law* $v(x) = \frac{1}{4\pi} \int \frac{\omega(y) \times |x-y|}{|x-y|^3} \mathrm{d}y$, which gives the velocity as a function of the vorticity, and thus makes clear the non-local interaction between the vortices.

In a similar manner, the *EPDiff equation* [HSSE09, Eq. 11.20] is given by

$$\underbrace{\frac{\partial m}{\partial t} + v \cdot \nabla m}_{\text{advection}} + \underbrace{\nabla v^T \cdot m + (\operatorname{div} v) m}_{\text{stretching}} = 0 \,, \tag{2}$$

together with the *non-penetration boundary condition* $v \cdot n = 0$ at the boundary of the domain, and it describes the motion of *wavefronts*, i.e., concentrations of linear momentum localized along hypersurfaces, each with its own direction and speed. Like the vorticity, the momentum m is advected by the velocity field v, while the stretching terms (with an extra $(\operatorname{div} v)m$ term because of compressibility) are necessary to ensure the *preservation of the (total) kinetic energy* $\frac{1}{2} \int v \cdot m \, \mathrm{d}x$.

As before, we need to close the system with a suitable constitutive relation between the velocity v and the momentum m. In most applications, where the fluid density ρ is constant, we essentially identify the momentum with the velocity; the Navier–Stokes for instance, although describing momentum transport as EPDiff does, are written exclusively in terms of the velocity. However, given the almost singular distribution of momentum in this case, attempting to self-advect with $v = m$ does not lead to a well-behaved flow. We take instead the *Kelvin-filtered velocity* (see e.g., [FHT01]):

$$v = D_\alpha^{-1} m = (I - \alpha^2 \Delta)^{-1} m \,, \tag{3}$$

where Δ is the vector Laplacian and α is a regularization parameter that regulates the typical thickness of the fronts. The non-local nature of the velocity field, and hence of the interaction between the wavefronts, becomes clearer if we rewrite it in the integral form $v(x) = \int G(x,y) m(y) \, \mathrm{d}y$ where $G(x,y)$ is the Green function of the elliptic operator D_α. In Fig. 6, we demonstrate the non-local relationship between the velocity and the momentum, by showing how initial conditions are typically set for varying values of α.

Our goal is to design a *stable* and *efficient* numerical scheme for EPDiff, and our guiding principle is to use an as-cheap-as-possible integrator which still satisfies the properties of the smooth equations: energy preservation, time-reversibility and the constitutive relationship between the momentum and the velocity. To do that, we re-formulate EPDiff in a way that makes it easier to see how the these properties arise in the smooth case. Then, we will use these insights to guide our choice of numerics.

First we note that Equation (2) can be written as:

$$\frac{\partial m}{\partial t} - R(v,m) = 0, \quad m = D_\alpha v \,, \tag{4}$$

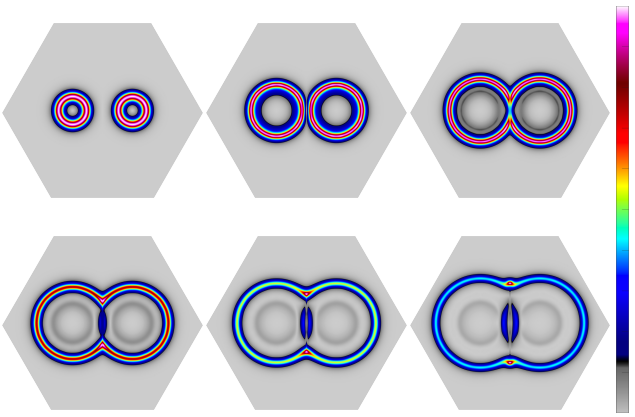

Figure 5: *Our method propagates momentum over velocity. We show the norm of the velocity v using the colorbar on the right. The obtained fronts exhibit a unique behavior. In particular, after collision, we obtain both the standard viscosity solution and additional fronts which appear in the periodic solution. See also Fig. 3.*

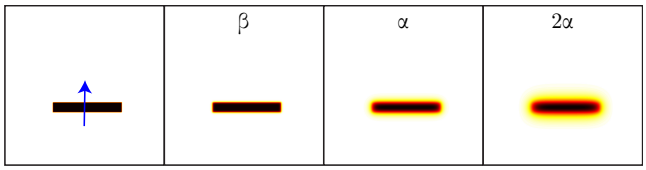

Figure 6: *Typical initial conditions. (left) A concentrated unit length momentum in black is smoothened with a vector Laplacian with parameter* β *yielding* m_0 *(left, middle). Then,* v_0 *is achieved by smoothening* m_0 *with parameter* α *(right, middle). Taking twice as large parameter leads to a smoother result (right). Notice that the direction of* m_0 *and* v_0 *is not affected by the smoothening, and it is shown with the blue arrow (left).*

with $R(v,m) = -\left(v\cdot\nabla m + \nabla v^T\cdot m + (\operatorname{div} v)m\right)$. This formulation allows us to focus on the two important operators R and D_α, and the properties they need to have for the smooth properties to hold. Specifically, we have that:

Lemma. Let $(m(t),v(t))$ be a solution of Equation (4). If D_α is a *self-adjoint* operator with respect to the L^2 inner product $\langle u,v\rangle := \int u\cdot v\,dx$, i.e., $\langle u, D_\alpha v\rangle = \langle D_\alpha u, v\rangle$, then $\frac{d}{dt}\frac{1}{2}\langle v,m\rangle = 0$.

Proof Since D_α is time-independent and self-adjoint, we get $m_t = D_\alpha v_t$ and $\langle v_t, m\rangle = \langle v, m_t\rangle$ where m_t is the time derivative of m, and so it follows that

$$\frac{d}{dt}\frac{1}{2}\langle v,m\rangle = \langle v,m_t\rangle\ .$$

Using Eq. (4), we have $\langle v,m_t\rangle = \langle v,R(v,m)\rangle$. Thus to complete the proof we need $\langle v,R(v,m)\rangle = 0$ for any v, which can be shown using standard vector calculus identities (see Appendix A). □

Thus, we have identified the two properties which are necessary for energy preservation, namely D_α should be self adjoint and $\langle v,R(v,m)\rangle = 0$ for any v. We will therefore assume these properties as given, and design a time discretization accordingly. We first propose a *structure-preserving integrator (SPI)*, solving separately for v and m, such that the resulting scheme is *time-invertible* at the price of allowing v and m to drift from their constitutive relation. We then show how this can be amended by giving up time-invertibility and proposing an *averaged structure-preserving integrator (ASPI)*. We also mention how all these properties can be achieved with an implicit scheme, at the expense of a higher computational burden (see Appendix B).

3. Discrete Setting

3.1. Time discretization

Structure-preserving integrator (SPI). We integrate the abstract form of the EPDiff equation as given in Eq. (4) using a *splitting scheme*. Our integrator is made up of successive substeps that update either the momentum or the velocity alone, and it eventually outputs the next (m^{k+1},v^{k+1}) at time $t^{k+1} = t^k + \tau$. While the proposed method preserves the energy and can be inverted in time, it unfortunately breaks the relation between m and v as is defined in Eq. (3). Thus, we introduce the notation $\bar v = D_\alpha^{-1}m$ and $\bar m = D_\alpha v$ to simplify notation and to better distinguish between the quantities m,v that we track in practice vs. their corresponding $\bar v,\bar m$. Our update rule is then

$$m^{k+\frac{1}{2}} = m^k + \frac{\tau}{2}R(v^k,\bar m^k)\ , \tag{5a}$$

$$v^{k+1} = v^k + \tau D_\alpha^{-1}R(\bar v^{k+\frac{1}{2}}, m^{k+\frac{1}{2}})\ , \tag{5b}$$

$$m^{k+1} = m^{k+\frac{1}{2}} + \frac{\tau}{2}R(v^{k+1},\bar m^{k+1})\ , \tag{5c}$$

where (m^k,v^k) are the momentum and velocity at time t^k. It is straightforward to check that the proposed scheme is *fully explicit* and *second order accurate in time*. The proposed form of "half step—entire step—half step" can be identified on the one hand as a *Strang splitting* [Str68], while on the other hand it closely mirrors the structure of the well-known *second order symplectic Verlet integrator* [Rut83] for separable Hamiltonian systems.

Energy preservation and time invertibility. We can show that each substep, and therefore the entire scheme, leaves the kinetic energy $E = \frac{1}{2}\langle v,m\rangle$ invariant, see the Appendix A for a proof. In Fig. 7 we show that indeed, energy is preserved in all our simulations up to machine precision. A second important property which Eq. (2) satisfies is *time invertibility*, namely that the transformation $m\to -m$, $v\to -v$ and $t\to -t$ leaves the constitutive relation (3) and the entire PDE invariant. This property also holds in the time discrete case when integrating with the update rule in Eqs. (5) in the following sense. Given a set of solutions $\{(m^i,v^i)\}$ computed using SPI, one can arrive at $(-m^k,-v^k)$ by starting from $(-m^{k+1},-v^{k+1})$ and applying the integrator. This can be shown for a substep and similarly for all steps by noticing that $-m^{k+\frac{1}{2}} = -m^{k+1} + \frac{\tau}{2}R(-v^{k+1},-\bar m^{k+1})$ since R is bilinear and D_α is linear. See Fig. 8 for an example of this property.

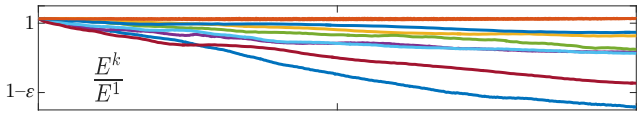

Figure 7: *The relative energy* E^k/E^1 *for all the simulations we demonstrate. Notice that the error is close to machine precision since* $\varepsilon = 2\times 10^{-13}$.

Second order convergence. Our splitting scheme voids the connection between velocities and momenta, i.e., v^{k+1} is *not* exactly the smoothened version of m^{k+1} and a certain drift occurs. Nevertheless, we confirm that our method is second order in time by measuring the drift and showing convergence in Fig. 9. We tried two different and complex scenarios on the sphere (red and blue) with exponentially decreasing time steps τ. We quantify the numerical error using the α norm $\|u\|_\alpha^2 := \langle u, D_\alpha u\rangle$:

$$\|e\|_\alpha = \|v - D_\alpha^{-1}m\|_\alpha = \langle v - D_\alpha^{-1}m, D_\alpha v - m\rangle^{\frac{1}{2}}\ ,$$

and, for comparison, show a quadratic function of τ (black).

Ideally, one would expect a numerical scheme which preserves all the properties that we listed, and, indeed, we discuss in Appendix B how to devise such a scheme, based on a mid-point rule,

that is fully implicit. However, this scheme is less practical since it involves a linear solve per step of a matrix which *depends* on the current step (compare with SPI which only inverts D_α). Instead, we show next a modified integrator that is again fully explicit, and offers a practical tradeoff – the energy and the constitutive relation $m = D_\alpha v$ are preserved, but time invertibility is no longer maintained.

Averaged structured-preserving integrator (ASPI). Using the integrator (5) we have at the end of a full step $v^{k+1} \neq D_\alpha^{-1} m^{k+1}$. One could attempt to rectify this by, for instance, discarding the computed velocity and taking the associated quantity $\bar{v}^{k+1} := D_\alpha^{-1} m^{k+1}$ instead. The problem is that, although this restores the constitutive relation, it breaks energy preservation since

$$\langle \bar{v}^{k+1}, m^{k+1} \rangle \neq \langle v^{k+1}, m^{k+1} \rangle = \langle v^k, m^k \rangle \, .$$

Interestingly, the tuple $(\bar{v}^{k+1}, m^{k+1})$ *overshoots* the energy, whereas the alternative tuple $(v^{k+1}, \bar{m}^{k+1})$ *undershoots* it (see the section 'Energy preservation for the ASPI' in Appendix A). Thus, since v^{k+1} and $\bar{v}^{k+1}$ are both estimates of the velocity at time t^{k+1}, we can look for a (step dependent) *convex combination* that blends these velocities and keeps the energy fixed. We therefore choose a scalar parameter $\lambda \equiv \lambda(\bar{v}^{k+1}, v^{k+1})$ as

$$\lambda(\bar{v}, v) := \left(1 + \left| \frac{\langle \bar{v}, \bar{v} - v \rangle}{\langle v, \bar{v} - v \rangle} \right|^{\frac{1}{2}} \right)^{-1} \leq 1 \, , \tag{6}$$

and compute the next velocity using $v_\lambda = (1-\lambda) v^{k+1} + \lambda \bar{v}^{k+1}$, with its associated momentum $m_\lambda := D_\alpha v_\lambda$. It turns out (see Appendix A) that when the ratio of the inner products is positive, then $\langle v_\lambda, m_\lambda \rangle = \langle v^k, m^k \rangle$ and the energy is preserved exactly. This is indeed the case under normal circumstances, since v and $\bar{v}$ are both approximations of $v(t^{k+1})$. If, on the other hand, the ratio is negative, we still have $\langle v_\lambda, m_\lambda \rangle = \langle v^k, m^k \rangle + \mathrm{O}(\tau^3)$.

3.2. Spatial discretization

Geometry, functions, vector fields and inner products. We consider a triangle mesh $\mathcal{M}$ where $\mathcal{V}, \mathcal{E}$ and $\mathcal{F}$ are its vertices, edges and faces, respectively. When required, we attach subscripts to

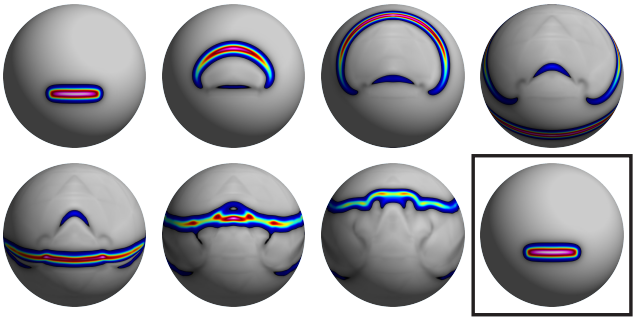

Figure 8: *Our SPI scheme is time invertible which allows to flow forward in time (top row and left bottom row), and then flow backwards and arrive at the initial conditions (black frame to the right).*

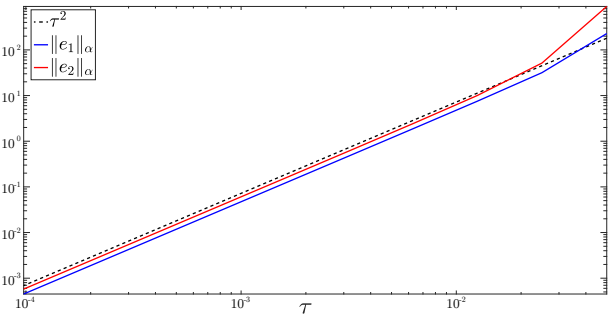

Figure 9: *Quadratic convergence in time. Error plots for two simulations (blue, red) where we measure how far v is from the smoothened m for decreasing time steps. For comparison, we show a quadratic function of τ (black). See the text for details.*

quantities to denote their domain, e.g., $f_\mathcal{V}$ is a function on vertices, and we similarly use $\mathcal{E}$ or $\mathcal{F}$. We use a typical setup of piecewise-linear functions and piecewise-constant vector fields, along with their associated inner products. Namely, we represent real-valued functions as scalars on the vertices of the mesh, i.e., $f \in \mathbb{R}^{|\mathcal{V}|}$, and extend them to the whole mesh using piecewise-linear hat basis functions. Notice that in this case gradients of functions are piecewise-constant, and thus in our setup vector fields are sampled using a constant tangent vector per face, i.e., $v \in \mathbb{R}^{3|\mathcal{F}|}$.

For defining mass matrices we require vertex, edge, and face areas, denoted by $a_\mathcal{V} \in \mathbb{R}^{|\mathcal{V}|}$, $a_\mathcal{E} \in \mathbb{R}^{|\mathcal{E}|}$ and $a_\mathcal{F} \in \mathbb{R}^{|\mathcal{F}|}$, respectively. Vertex areas are $1/3$ of the total area of their adjacent triangles, and similarly, we use $1/3$ of the sum of area of the neighboring triangles for edge area. The resulting diagonal mass matrices are denoted by $G_\mathcal{V} \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$, $G_\mathcal{E} \in \mathbb{R}^{|\mathcal{E}|\times|\mathcal{E}|}$, and $G_\mathcal{F} \in \mathbb{R}^{|\mathcal{F}|\times|\mathcal{F}|}$ for vertices, edges and faces, respectively. Additionally, we use $\langle , \rangle$ to denote L^2 inner products; for example the inner product between functions on vertices is given by $\langle f_\mathcal{V}, g_\mathcal{V} \rangle = f_\mathcal{V}^T G_\mathcal{V} g_\mathcal{V}$. For L^2 inner products between vector fields on the faces and on the vertices, we define $G_{3\mathcal{F}}, G_{3\mathcal{V}}$ by repeating the corresponding mass matrices 3 times. This yields, e.g., the discrete inner product $\langle v, m \rangle = v^T G_{3\mathcal{F}} m$.

We define a matrix $I_\mathcal{V}^\mathcal{F} \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{F}|}$ which interpolates quantities from faces to vertices, i.e., $I_\mathcal{V}^\mathcal{F}(i,j) = (1/3) a_\mathcal{F}(j) / a_\mathcal{V}(i)$, iff vertex i belongs to face j and 0 otherwise. Similarly, $I_\mathcal{F}^\mathcal{V}$ interpolates data from vertices to faces and is defined by $I_\mathcal{F}^\mathcal{V} = G_\mathcal{F}^{-1} (I_\mathcal{V}^\mathcal{F})^T G_\mathcal{V}$, such that $\langle f_\mathcal{F}, I_\mathcal{F}^\mathcal{V} g_\mathcal{V} \rangle = \langle I_\mathcal{V}^\mathcal{F} f_\mathcal{F}, g_\mathcal{V} \rangle$ holds. To reduce clutter we sometimes denote an interpolated quantity $f_\mathcal{V} = I_\mathcal{V}^\mathcal{F} f_\mathcal{F}$ by $\tilde{f}$.

Differential Operators. To preserve energy discretely, we need the operators grad $\in \mathbb{R}^{3|\mathcal{F}|\times|\mathcal{V}|}$ and div $\in \mathbb{R}^{|\mathcal{V}|\times 3|\mathcal{F}|}$ to fulfill discrete integration by parts, thus we define div $= -G_\mathcal{V}^{-1}$ grad^T $G_{3\mathcal{F}}$. This definition coincides with the standard construction of these operators (e.g., as defined in [BKP*10, Chapter 3]). To construct D_α we further need a vectorial Laplacian. Instead of using the scalar Laplacian component-wise, we use the following *Hodge Laplacian* [dGDT15] suited for curved domains

$$\Delta = G_{3\mathcal{F}}^{-1} \left(\mathrm{div}^T \, G_\mathcal{V} \, \mathrm{div} + J^T \, \mathrm{div}_\mathcal{E}^T \, G_\mathcal{E} \, \mathrm{div}_\mathcal{E} \, J \right) \, ,$$

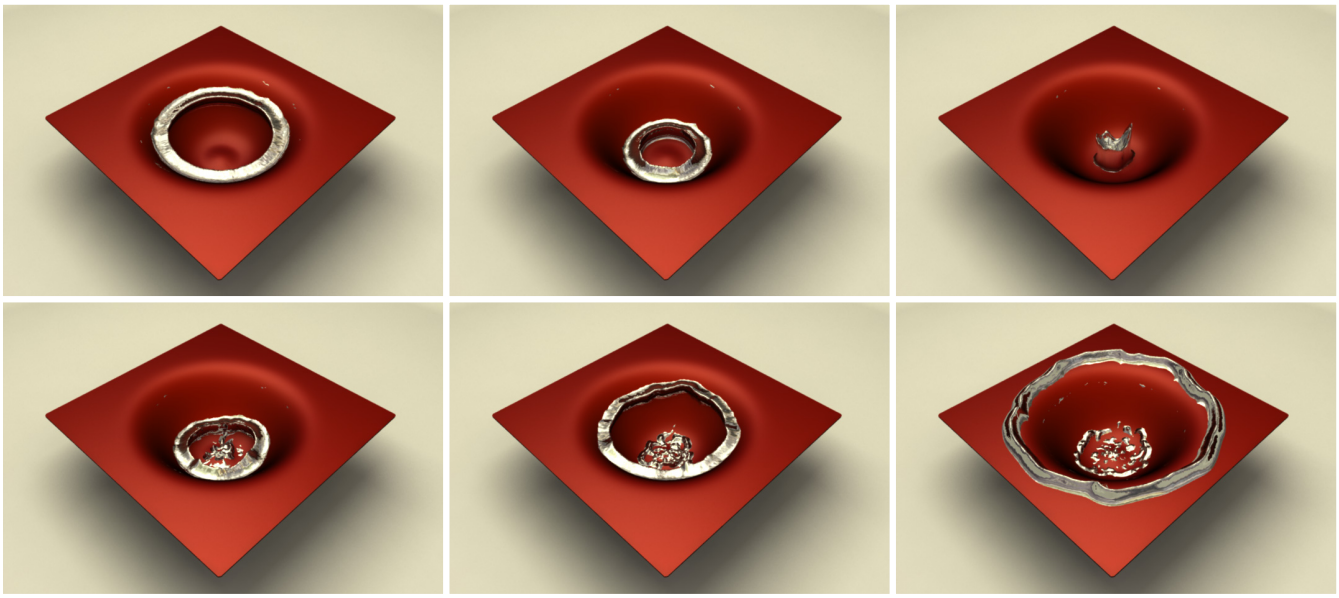

Figure 10: *A circular front is shrinking towards the center of the bowl till it collapses, then the front re-emerges and propagates outward.*

where J is an operator which rotates tangent vectors with respect to the face normal, and $\mathrm{div}_{\mathcal{E}}$ is the edge-based divergence operator. It is easy to verify that Δ is self-adjoint with respect to the inner product since the operator in the parentheses can be written as a multiplication between a matrix and its transpose.

Finally, we need an operator $\nabla\tilde{m}$ which provides the Jacobian of a vertex-based vector field $\tilde{m}$, at the j-th face. Thus we define $(\nabla\tilde{m})_j = \left((\mathrm{grad}\,\tilde{m}_x)_j\,(\mathrm{grad}\,\tilde{m}_y)_j\,(\mathrm{grad}\,\tilde{m}_z)_j\right)^T \in \mathbb{R}^{3\times 3}$.

Discrete EPDiff equation. Using the above notation and discrete operators, we can discretize $R(v,m)$ as follows:

$$R(v,m)_j := -\Big\{(\nabla\tilde{m})_j v_j - (\nabla\tilde{m})_j^T v_j + (\mathrm{grad}(\tilde{m}\cdot\tilde{v}))_j + \sum_i (I_{\mathcal{F}}^{\mathcal{V}})_{ji}(\mathrm{div}\, v)_i \tilde{m}_i\Big\}\,. \quad (7)$$

Comparing with the EPDiff equation (2), this is a rather direct discretization, additionally using necessary interpolations between face- and vertex-based quantities (see also the section 'Energy Preservation for the SPI' in Appendix A). As we have seen in the previous sections, *the kinetic energy is preserved as long as* $\langle v, R(v,m)\rangle = 0$ *for all (discrete) velocities* v. Indeed, we note that $\langle v, R(v,m)\rangle = \sum_j a_{\mathcal{F}}(j) v_j^T R(v,m)_j$ and the contributions of the first two terms in (7) cancel out, since $v_j^T(\nabla\tilde{m})_j v_j = (v_j^T(\nabla\tilde{m})_j v_j)^T = v_j^T(\nabla\tilde{m})_j^T v_j$. Likewise, the contribution of the last two terms also cancel out. In the smooth setting, integration by parts leads to $\langle v, \mathrm{grad}(m\cdot v)\rangle = -\langle v, (\mathrm{div}\, v)m\rangle$. Similarly, using our discrete operators, it is straightforward to show that $\langle v, \mathrm{grad}(\tilde{m}\cdot\tilde{v})\rangle = -\left\langle v, I_{\mathcal{F}}^{\mathcal{V}}(\mathrm{div}\, v)\tilde{m}\right\rangle$. Furthermore, the resulting scheme is *time-invertible*, as it is easy to check that $R(-v,-m) = R(v,m)$.

4. Implementation details

We implemented our technique in MATLAB. Given initial m_0 and its corresponding v_0, at each step we integrate in time using our SPI or ASPI methods by computing the discrete $R(v,m)$ which appears in Eq (7), evaluating the discrete D_α and facilitating its pre-computed factorization for solving the required linear systems.

The initial momentum is set by prescribing a delta function over the faces which represent the front, and smoothening it using a small parameter β. This provides the norm of m_0 and its direction was commonly chosen as the normalized gradient of a distance function. Then, the initial velocity v_0 is computed by the non-local relation, see also Fig. 6. A rule of thumb for choosing α is to use 3-4 times the average edge length of the mesh. We provide the parameters α and β for our simulations in Table 1.

Per step, the inverse of the non-local D_α matrix is needed twice in SPI and three times in ASPI. Fortunately, this matrix depends only on the mesh and is independent from the entire simulation. Thus, we pre-factor it once as a pre-processing step using the SuiteSparse library [Dav06]. With the pre-factorization in place, the estimate for the computational cost per step can be roughly divided to 60% for evaluations of R and 40% for the linear solves for SPI and 50%/50% for ASPI. We show further details and timings of our simulations in Table 1.

Finally, we employ a basic Courant–Friedrichs–Lewy (CFL) rule for the dynamic time stepping. Specifically, the next time step τ^{k+1} is updated using the relation

$$\tau^{k+1} = \gamma\,\delta x / \|v^k\|_\infty\,,$$

where γ is a fixed constant in all of our simulations taken to be 0.4 for SPI and 0.8 for ASPI, and the average edge length δx is divided

by the infinity norm of the current velocity, so that propagation distance is limited to at most a single cell per step.

Limitations. Our method has a few inherent disadvantages which suggest future exploration. For instance, our model makes use of the averaged velocity per fluid column, thus effects that result from different velocities, e.g., breaking of waves, are not possible. Another shortcoming of our method is its Eulerian nature, enforcing the usage of highly resolved triangle meshes with relatively small time steps. However, since the simulated fronts are extremely concentrated, one might consider to extend our method to incorporate adaptive techniques and we leave it for future investigation.

5. Results

Self-collapsing circular front. The initial circular front is propagating quickly towards the center of the bowl till it collapses into a small region (Fig 10, top row). The visible amount of fluid is preserved in this experiment exactly due to energy conservation which is crucial in such stress tests. However, the velocity-momentum relation must be also maintained in such extreme cases. For comparison, the SPI method stops (τ becomes zero) at the collapsing point, whereas the ASPI method passes this point, allowing the circular front to re-emerge and flow outward (Fig 10, bottom row).

Newton's cradle on a torus. In Fig. 11, three rings of different momenta are placed on the torus where the ring with the higher momentum travels faster than the other two rings. When two rings collide, an exchange of momentum occurs, similar to the 1D case shown in Fig. 2. Thus, this experiment mimics the rigid body scenario where there is a perfect exchange of momentum between swinging spheres. Notice that while momentum is not perfectly exchanged in our simulation, the general shape and intensity of the rings are preserved.

Multiple fronts on a curved geometry. We can easily set multiple fronts in the same simulation allowing them to interact between themselves. Additionally, we show that the left front exhibits a different profile compared to the right front due to the underlying bump (Fig. 12, top row, left). Then, the fronts collide and secondary waves naturally emerge (Fig. 12, top row, middle and right). Later, the concentrated waves continue to propagate and interact with the scene in a plausible manner (12, bottom row).

Flow behind obstacles. Fig. 15, top left shows how a concentrated wave breaks when it hits the cylinder. The front then passes the obstacle and reconnects afterwards, see Fig. 15, top right. It continues to sweep through the pool in a natural way, while additional fronts are generated from behind the cylinder, see Fig. 15, bottom.

6. Wavefront-driven globally-supported periodic waves

In this section we show how our approach can be easily extended to support periodic wave functions in a post-processing step. To this end, we employ a basic version of a wavefront tracking method (see e.g., [JW15]) where our simulated concentrated wave replaces the traditional tracked front resulting from solving the eikonal eq. Specifically, we stack the interpolated velocity norms onto a matrix $V \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{T}|}$, i.e., each column is given by $V_j = I_{\mathcal{V}}^{\mathcal{F}}|v_j|$, where $|v_j| \in \mathbb{R}^{|\mathcal{F}|}$ is the point-wise norm of the velocity at time $t_j \in \mathcal{T}$, where $\mathcal{T} = \{0,\dots,t_{\max}\}$ is the set of discrete times that we have data for. Interestingly, the non-zero entries of the matrix V exactly identify the locations of our concentrated wave over time. Thus, each non-zero V_{ij} represents that the front visited node x_i at time t_j. Notice that due to the nature of the EPDiff eq., the concentrated fronts are attenuated over time, therefore, in practice, we modify the entries by applying a gamma correction, $(V_\gamma)_{ij} = V_{max}(V_{ij}/V_{max})^\gamma, \gamma < 1$.

What is left to determine is the correct amplitude per node at each time t. We assume that the source of the wavefront starts oscillating at time $t = 0$ with an amplitude that is a sum of N sinusoidal modes

$$\phi(t) = \sum_{k=1}^{N} \phi_k(t) = \sum_{k=1}^{N} a_k \sin(\omega_k \max(t,0)),$$

with amplitudes a_k and (angular) frequencies ω_k. The amplitude then at node x_i sampled at a time t, can be written as a convolution of a) the velocities V_{ij} at the node at various times t_j with b) the *time-delayed* amplitude of the source $\phi(t-t_j)$. As mentioned, every non-zero velocity entry V_{ij} implies that a new wavefront arrives at node x_i at time t_j, bringing with it a time-shifted (and potentially attenuated) copy of the source signal. Taking into account the extra effect of *(shallow/deep) water dispersion*, where different frequencies propagate at different speeds [JW15], we arrive at the following sum over modes ϕ_k and times t_j:

$$\eta(x_i,t) = \sum_{k=1}^{N} \sum_{t_j \in \mathcal{T}} (V_\gamma)_{ij} \phi_k(b_k t - t_j) \tag{8}$$

The parameters $b_k = b(\omega_k)$ implement the dependency of the speed of the sinusoidal waves (as a fraction of the fixed pre-calculated speed of the wavefront) on the frequency. In Fig. 13 we demonstrate a graphical interpretation of the proposed method, where a 1D wave is traveling and bouncing off the boundary of the domain over time (black line). The front data is then coupled with two wave sources propagating at different speeds and different frequencies (blue and red curves shown on top). The resulting convolution (black vertical curve to the right) is the sum of the separate convolutions (red and blue vertical curves).

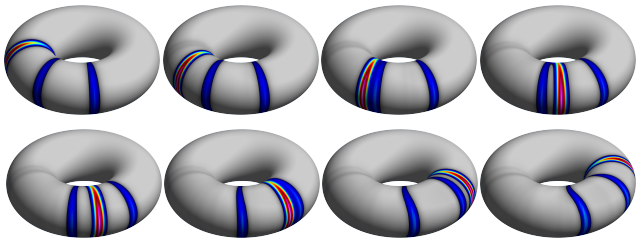

Figure 11: *Newton's cradle on a torus with rings of different momenta. Upon collision, an exchange of momentum occurs allowing the rings to "switch" places while roughly maintaining their original shape and intensity. Fig 2 shows a similar experiment in 1D.*

Figure	$\lvert\mathcal{V}\rvert$	δx	α	β	#steps	Average time per step	Total time
Fig. 4, Bunny	166427	0.0027	0.01	1e-5	401	4.095	1642.1
Fig. 5, Circular fronts	66141	0.003	0.015	1.5e-5	101	0.879	88.7
Fig. 8, Time reversibility	40962	0.019	0.05	5e-5	957	0.554	531.1
Fig. 10, Mercury on a bowl	50876	0.005	0.015	1.5e-5	381	0.843	321.5
Fig. 11, Newton's cradle on a torus	40000	0.009	0.015	1.5e-4	191	0.685	131
Fig. 12, Ocean	43676	0.005	0.017	5.6e-5	286	0.688	196.9
Fig. 14, Periodic circular wave	66141	0.003	0.015	1.5e-5	141	0.879	124
Fig. 15, Pool	39990	0.005	0.018	1.8e-5	201	0.383	77
Fig. 16, Atlas	79952	0.0025	0.01	0	786	0.758	595.8

Table 1: *We show various statistics for our simulations including the number of vertices $|\mathcal{V}|$, the average edge length δx, the smoothening parameters α, β, the number of steps and computation times in seconds.*

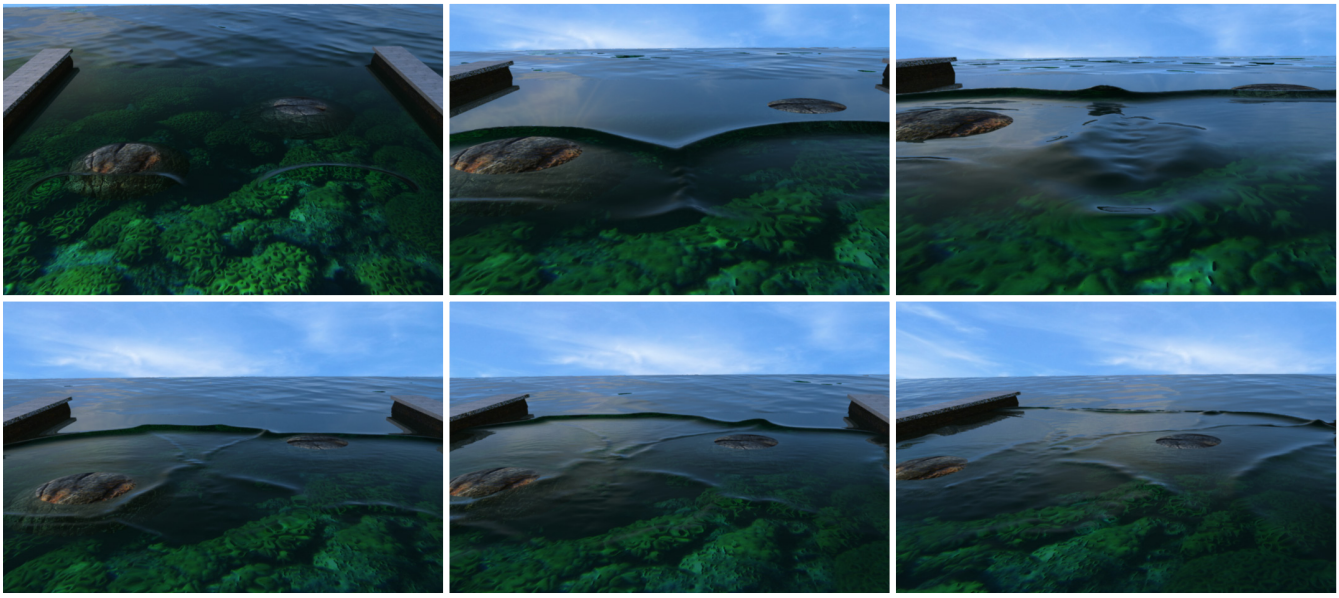

Figure 12: *Multiple fronts on a curved domain propagate with different profiles due to the underlying geometry (top row, left). When the fronts meet, additional concentrated waves appear (top row, middle and right). The general behavior is realistic showing wave-wave and wave-scene interactions (bottom row).*

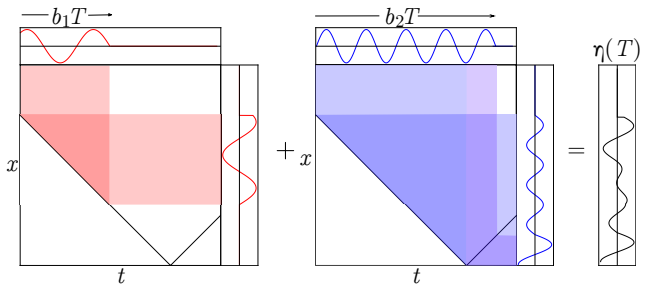

Figure 13: *To support periodic waves, we convolve the tracked front (black line) with wave sources having different frequencies and propagation speeds (red and blue curves shown on top). The resulting periodic wave is obtained from the sum of the two convolutions (black curve to the right).*

In practical terms, the calculation of the vector $\eta(t)$ of the amplitude of all the nodes x_i at a given time t can be performed efficiently as the matrix-matrix-vector product:

$$\eta(t) = V_\gamma \Phi(t)\, a \tag{9}$$

where V_γ is the gamma-corrected wavefront velocity matrix, the columns of $\Phi(t) \in \mathbb{R}^{|\mathcal{T}|\times N}$ are the time-shifted sinusoidal modes sampled at the times $\mathcal{T}$ of the form $\sin(\omega_k \max(b_k t - \mathcal{T}, 0))$, and $a = (a_1 \dots a_N)^T$ is the vector of the amplitudes of the various source modes. Note that the generated wave $\eta(t)$ is a *piecewise-linear* function on vertices which is sufficiently sampled in time and space due to our CFL condition (see Section 4), thus linear interpolation in space produces reasonable results. In Fig. 14 we show how a single circular front (left) can be convolved with a single periodic source to produce a periodic behavior (middle). Also, coupling with two sources traveling at different speeds, allows to achieve dispersion-like effects (right).

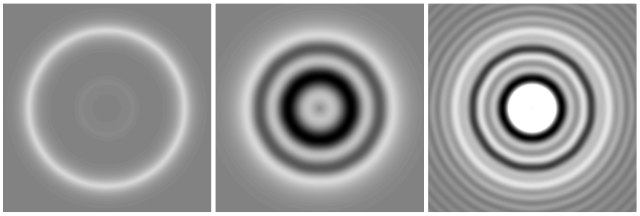

Figure 14: *A circular front (left) is coupled with a single shifted source, producing a periodic wave (middle). Using the same technique, several sources can be blended, achieving dispersion effects.*

7. Conclusion

We proposed a novel numerical scheme for the EPDiff equation while maintaining geometrical invariants. We tested our approach in the context of simulating concentrated and globally-supported waves on general triangle meshes. Moreover, we presented two numerical splitting schemes which are fully explicit and conserve the energy by construction, allowing to choose time invertibility over preservation of the velocity-momentum relation and vice versa. Our method is extremely efficient since it involves at most three inversions of a fixed matrix which we factorize as a pre-processing step. Additionally, we showed that our numerical data can be used in the context of wavefront tracking methods where several linear wave sources are blended to obtain various wave effects such as periodicity and dispersion. We demonstrated interesting wave-wave and wave-scene interaction effects on flat and curved domains.

We believe that many potential extensions are worth exploring. In particular, adding various forcing terms to the governing PDE will probably yield intricate new effects. Also, extending the technique to support adaptive meshes will be useful, since large parts of the domain do not contain any parts of the resulting front, and thus the resolution of these parts can be reduced. Finally, allowing layers of velocities instead of a single tangential vector might be an interesting direction as it will allow to achieve a richer span of motion effects such as breaking of waves.

Acknowledgments. We would like to thank Eric Yudin for his help with rendering the figures in this work. The first author acknowledges an Adams Fellowship. This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme under grant agreement No 714776.

References

[AVBC16] AZENCOT O., VANTZOS O., BEN-CHEN M.: Advection-based function matching on surfaces. In *Computer Graphics Forum* (2016), vol. 35, Wiley Online Library, pp. 55–64. 3

[AVW*15] AZENCOT O., VANTZOS O., WARDETZKY M., RUMPF M., BEN-CHEN M.: Functional thin films on surfaces. In *Proceedings of the 14th ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2015), ACM, pp. 137–146. 1

[AWO*14] AZENCOT O., WEISSMANN S., OVSJANIKOV M., WARDETZKY M., BEN-CHEN M.: Functional fluids on surfaces. In *Computer Graphics Forum* (2014), vol. 33, Wiley Online Library, pp. 237–246. 1

[BKP*10] BOTSCH M., KOBBELT L., PAULY M., ALLIEZ P., LÉVY B.: *Polygon mesh processing*. CRC press, 2010. 6

[BMTY05] BEG M. F., MILLER M. I., TROUVÉ A., YOUNES L.: Computing large deformation metric mappings via geodesic flows of diffeomorphisms. *International journal of computer vision 61*, 2 (2005), 139–157. 3

[Bri08] BRIDSON R.: *Fluid simulation for computer graphics*. CRC Press, 2008. 2, 3

[CKL14] CAMASSA R., KUANG D., LEE L.: Solitary waves and *n*-particle algorithms for a class of euler-poincar\'e equations. *arXiv preprint arXiv:1404.4858* (2014). 3

[CM10] CHENTANEZ N., MÜLLER M.: Real-time simulation of large bodies of water with small scale details. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics symposium on computer animation* (2010), Eurographics Association, pp. 197–206. 3

[CTM12] CHERTOCK A., TOIT P. D., MARSDEN J. E.: Integration of the epdiff equation by particle methods. *ESAIM: Mathematical Modelling and Numerical Analysis 46*, 03 (2012), 515–534. 3

[Dav06] DAVIS T. A.: *Direct methods for sparse linear systems*, vol. 2. Siam, 2006. 7

[DCGG11] DARLES E., CRESPIN B., GHAZANFARPOUR D., GONZATO J.-C.: A survey of ocean simulation and rendering techniques in computer graphics. In *Computer Graphics Forum* (2011), vol. 30, Wiley Online Library, pp. 43–60. 3

[DD91] DALRYMPLE R. A., DEAN R. G.: *Water wave mechanics for engineers and scientists*. Prentice-Hall, 1991. 2

[dGDT15] DE GOES F., DESBRUN M., TONG Y.: Vector field processing on triangle meshes. In *SIGGRAPH Asia 2015 Courses* (2015), ACM, p. 17. 6

[FF01] FOSTER N., FEDKIW R.: Practical animation of liquids. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques* (2001), ACM, pp. 23–30. 3

[FHT01] FOIAS C., HOLM D. D., TITI E. S.: The navier–stokes-alpha model of fluid turbulence. *Physica D: Nonlinear Phenomena 152* (2001), 505–519. 4

[FR86] FOURNIER A., REEVES W. T.: A simple model of ocean waves. In *ACM Siggraph Computer Graphics* (1986), vol. 20, ACM, pp. 75–84. 3

[GLS00] GONZATO J.-C., LE SAËC B.: On modelling and rendering ocean scenes. *The Journal of Visualization and Computer Animation 11*, 1 (2000), 27–37. 3

[GM02] GAMITO M. N., MUSGRAVE F. K.: An accurate model of wave refraction over shallow water. *Computers & Graphics 26*, 2 (2002), 291–307. 3

[HLW06] HAIRER E., LUBICH C., WANNER G.: *Geometric numerical integration: structure-preserving algorithms for ordinary differential equations*, vol. 31. Springer Science & Business Media, 2006. 1

[HM05] HOLM D. D., MARSDEN J. E.: Momentum maps and measure-valued solutions (peakons, filaments, and sheets) for the epdiff equation. In *The breadth of symplectic and Poisson geometry*. Springer, 2005, pp. 203–235. 3

[HNC02] HINSINGER D., NEYRET F., CANI M.-P.: Interactive animation of ocean waves. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation* (2002), ACM, pp. 161–166. 3

[HS13] HOLM D. D., STALEY M. F.: Interaction dynamics of singular wave fronts. *arXiv preprint arXiv:1301.1460* (2013). 1, 2, 3

[HSSE09] HOLM D. D., SCHMAH T., STOICA C., ELLIS D. C.: *Geometric mechanics and symmetry: from finite to infinite dimensions*. Oxford University Press London, 2009. 3, 4

[JW15] JESCHKE S., WOJTAN C.: Water wave animation via wavefront parameter interpolation. *ACM Transactions on Graphics (TOG) 34*, 3 (2015), 27. 3, 8

[KB14] KEELER T., BRIDSON R.: Ocean waves animation using boundary integral equations and explicit mesh tracking. In *ACM SIGGRAPH 2014 Posters* (2014), ACM, p. 11. 3

[KM90] KASS M., MILLER G.: Rapid, stable fluid dynamics for computer graphics. In *ACM SIGGRAPH Computer Graphics* (1990), vol. 24, ACM, pp. 49–57. 3

[LMH*15] LIU B., MASON G., HODGSON J., TONG Y., DESBRUN M.: Model-reduced variational fluid simulation. *ACM Transactions on Graphics (TOG) 34*, 6 (2015), 244. 1

[LMMM16] LARSSON S., MATSUO T., MODIN K., MOLTENI M.: Discrete variational derivative methods for the epdiff equation. *arXiv preprint arXiv:1604.06224* (2016). 3

[MCP*09] MULLEN P., CRANE K., PAVLOV D., TONG Y., DESBRUN M.: Energy-preserving integrators for fluid animation. In *ACM Transactions on Graphics (TOG)* (2009), vol. 28, ACM, p. 38. 1

[MTY02] MILLER M. I., TROUVÉ A., YOUNES L.: On the metrics and euler-lagrange equations of computational anatomy. *Annual review of biomedical engineering 4*, 1 (2002), 375–405. 1, 3

[MWM*87] MASTIN G., WATTERBERG P., MAREDA J. F., ET AL.: Fourier synthesis of ocean scenes. *Computer Graphics and Applications, IEEE 7*, 3 (1987), 16–23. 3

[NB11] NIELSEN M. B., BRIDSON R.: Guide shapes for high resolution naturalistic liquid simulation. In *ACM Transactions on Graphics (TOG)* (2011), vol. 30, ACM, p. 83. 3

[NSB13] NIELSEN M. B., SÖDERSTRÖM A., BRIDSON R.: Synthesizing waves from animated height fields. *ACM Transactions on Graphics (TOG) 32*, 1 (2013), 2. 3

[Pea86] PEACHEY D. R.: Modeling waves and surf. In *ACM Siggraph Computer Graphics* (1986), vol. 20, ACM, pp. 65–74. 3

[Rut83] RUTH R. D.: A canonical integration technique. *Nuclear Science, IEEE Transactions on 30*, 4 (Aug 1983), 2669–2671. 5

[Saf92] SAFFMAN P. G.: *Vortex dynamics*. Cambridge university press, 1992. 2

[Str68] STRANG G.: On the construction and comparison of difference schemes. *SIAM Journal on Numerical Analysis 5*, 3 (1968), 506–517. 5

[T*01] TESSENDORF J., ET AL.: Simulating ocean water. *Simulating Nature: Realistic and Interactive Techniques. SIGGRAPH 1*, 2 (2001), 5. 3

[TB87] TS'O P. Y., BARSKY B. A.: Modeling and rendering waves: wave-tracing using beta-splines and reflective and refractive texture mapping. *ACM Transactions on Graphics (TOG) 6*, 3 (1987), 191–214. 3

[Tes04] TESSENDORF J.: Interactive water surfaces. *Game Programming Gems 4* (2004), 265–274. 3

[TMFSG07] THÜREY N., MÜLLER-FISCHER M., SCHIRM S., GROSS M.: Real-time breaking waves for shallow water simulations. In *Computer Graphics and Applications, 2007. PG'07. 15th Pacific Conference on* (2007), IEEE, pp. 39–46. 3

[Vre13] VREUGDENHIL C. B.: *Numerical methods for shallow-water flow*, vol. 13. Springer Science & Business Media, 2013. 2

[YHK07] YUKSEL C., HOUSE D. H., KEYSER J.: Wave particles. In *ACM Transactions on Graphics (TOG)* (2007), vol. 26, ACM, p. 99. 3

[ZBG15] ZHANG X., BRIDSON R., GREIF C.: Restoring the missing vorticity in advection-projection fluid solvers. *ACM Transactions on Graphics (TOG) 34*, 4 (2015). 4

Appendix A: Energy preservation and time invertibility

Energy Preservation for the SPI. The key technical lemma that we need for the energy preservation of the integrator (5), is the following orthogonality result:

Let $R(v,m) := -\Big(v\cdot\nabla m+\nabla v^T\cdot m+(\operatorname{div} v)m\Big)$. *Then* $\langle v,R(v,m)\rangle = 0$ *for any* v,m.

Proof Noting that $\nabla v^T\cdot m = m\cdot\nabla v + m\times(\nabla\times v)$, we can show that $R(v,m) = -\nabla(v\cdot m)+v\times(\nabla\times m)-(\operatorname{div} v)m$, a known alternate formulation for the EPDiff. Testing this expression with the velocity v, we get

$$\begin{aligned}&\langle v,R(v,m)\rangle\\ &\quad=-\langle v,\nabla(v\cdot m)\rangle+\langle v,v\times(\nabla\times m)\rangle-\langle v,(\operatorname{div} v)m\rangle\\ &\quad=-\langle v,\nabla(v\cdot m)\rangle-\langle v,(\operatorname{div} v)m\rangle\end{aligned}$$

where we immediately cancelled the middle term, since by the triple product identity $a\cdot(b\times c)=c\cdot(a\times b)$ we have $v\cdot(v\times(\nabla\times m))=(\nabla\times m)\cdot(v\times v)=0$.
We need to integrate the first term by parts. Integrating the vector calculus identity $\operatorname{div}(fv)=f\operatorname{div} v+\nabla f\cdot v$ with $f=v\cdot m$, and using the divergence theorem, gives us

$$\int_\Omega v\cdot\nabla(v\cdot m)=\int_{\partial\Omega}(v\cdot m)(v\cdot n)-\int_\Omega(\operatorname{div} v)(v\cdot m)$$

By the standard *no-penetration boundary condition* $v\cdot n=0$ the boundary integral on the RHS vanishes. It follows that $\langle v,\nabla(v\cdot m)\rangle=-\langle v,(\operatorname{div} v)m\rangle$, and so indeed $\langle v,R(v,m)\rangle=0$. □

The lemma can be used to show the following result:

If D_α *is self-adjoint and* $\langle v,R(v,m)\rangle = 0$ *for any* v, *then* $\langle v^k,m^k\rangle=\langle v^{k+1},m^{k+1}\rangle$.

Proof Let R^k, $R^{k+\frac{1}{2}}$ and R^{k+1} denote the three $R(\cdot,\cdot)$ terms in the three steps of the integrator (5) respectively. The property $\langle v,R(v,m)\rangle=0$ implies immediately that $\langle v^k,R^k\rangle=\langle v^{k+\frac{1}{2}},R^{k+\frac{1}{2}}\rangle=\langle v^{k+1},R^{k+1}\rangle=0$. It follows that every momentum

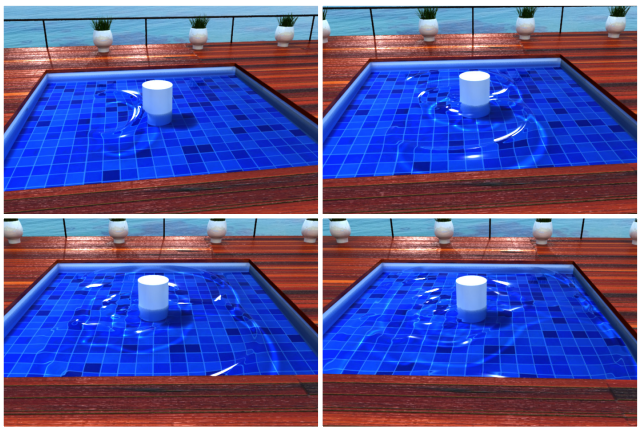

Figure 15: *Our method allows the concentrated waves to interact with obstacles in the scene in a natural way. The propagated front splits into two due to the cylinder and reconnects afterwards. Notice that additional waves are generated from behind the cylinder due to the no-penetration boundary conditions that our differential operators maintain.*

update is in an orthogonal direction to the associated velocity:

$$\begin{aligned}&\langle m^{k+1}, v^{k+1}\rangle = \langle m^{k+\frac{1}{2}} + \frac{\tau}{2}R^{k+1}, v^{k+1}\rangle \\ =&\langle m^{k+\frac{1}{2}}, v^{k+1}\rangle = \langle m^{k+\frac{1}{2}}, v^{k} + \tau D_\alpha^{-1} R^{k+\frac{1}{2}})\rangle \\ =&\langle m^{k+\frac{1}{2}}, v^{k}\rangle = \langle m^{k} + \frac{\tau}{2}R^{k}, v^{k}\rangle \\ =&\langle m^{k}, v^{k}\rangle\end{aligned}$$

Note that we used the fact that D_α is self-adjoint (and so D_α^{-1} too), together with the definition $\bar{v} = D_\alpha^{-1} m$, to eliminate the term $\langle m^{k+\frac{1}{2}}, D_\alpha^{-1} R^{k+\frac{1}{2}}\rangle = \langle D_\alpha^{-1} m^{k+\frac{1}{2}}, R^{k+\frac{1}{2}}\rangle = \langle \bar{v}^{k+\frac{1}{2}}, R^{k+\frac{1}{2}}\rangle$. □

Time invertibility for the SPI. *Let (m^k, v^k) and (m^{k+1}, v^{k+1}) satisfy the equations (5), and assume that $R(-v,-m) = R(v,m)$ for all v, m. Then starting with $(\tilde{m}^k, \tilde{v}^k) = (-m^{k+1}, -v^{k+1})$ and applying the integrator, we arrive at $(\tilde{m}^{k+1}, \tilde{v}^{k+1}) = (-m^k, -v^k)$.*

Proof Starting from $\tilde{v}^k = -v^{k+1}$ and using the assumption on R, we can verify that $\tilde{R}^k = R^{k+1}$, and so $\tilde{m}^{k+\frac{1}{2}} = \tilde{m}^k + \frac{\tau}{2}\tilde{R}^k = -m^{k+1} + \frac{\tau}{2}R^{k+1} = -m^{k+\frac{1}{2}}$. In the same manner we can follow the integrator steps backwards, flipping the signs of m and v on the way, to arrive at $\tilde{m}^{k+1} = -m^k$. □

Energy preservation for the ASPI. Consider the integrator (5), under the assumption that $m^k = D_\alpha v^k$, and in addition let $\bar{v}^{k+1} := D_\alpha^{-1} m^{k+1}$. If we take $v_\lambda := (1-\lambda)v^{k+1} + \lambda \bar{v}^{k+1}$ and $m_\lambda := D_\alpha v_\lambda$ with λ as in Eq (6), then

1. If $\langle v^{k+1}, Q\rangle \langle \bar{v}^{k+1}, Q\rangle \geq 0$ then $\langle v_\lambda, m_\lambda\rangle = \langle v^k, m^k\rangle$.
2. In any case, as long as $R(\cdot,\cdot)$ is bilinear, $\langle v_\lambda, m_\lambda\rangle = \langle v^k, m^k\rangle + O(\tau^3)$.

where $Q := \bar{v}^{k+1} - v^{k+1}$.

Proof Noting that the product of a velocity v with its associated momentum $m = D_\alpha v$ is equal to (the square of) its α norm, $\langle v, m\rangle = \|v\|_\alpha^2$, and making good use of the chain of equalities in the proof of the energy preservation of the integrator (5) (see above), we can show that

$$\begin{aligned}\|\bar{v}^{k+1}\|_\alpha^2 - \|v^k\|_\alpha^2 &= \tau\langle \bar{v}^{k+1}, Q\rangle \\ \|v^{k+1}\|_\alpha^2 - \|v^k\|_\alpha^2 &= -\tau\langle v^{k+1}, Q\rangle\end{aligned}$$

1. Using these, the desired condition $\|v_\lambda\|_\alpha^2 = \|v^k\|_\alpha^2$ is eventually equivalent to $\lambda^2\langle \bar{v}^{k+1}, Q\rangle = (1-\lambda)^2\langle v^{k+1}, Q\rangle$. If the two inner products have the same sign, then we can solve this quadratic for λ. The lambda given in (6) is the root in $[0,1]$.
2. On the other hand, if $R(\cdot,\cdot)$ is bilinear, then we can go back to the definition of the integrator and derive expansions of the form $R^{k+\frac{1}{2}} = R^k + \frac{\tau}{2}\ldots$ and $R^{k+1} = R^k + \tau\ldots$, where the coefficients are functions of v^k, m^k alone. From these we can check that $Q = O(\tau^2)$, and so the relations above can be read as $\|\bar{v}^{k+1}\|_\alpha^2 = \|v^k\|_\alpha^2 + O(\tau^3)$ and $\|v^{k+1}\|_\alpha^2 = \|v^k\|_\alpha^2 + O(\tau^3)$. Keeping in mind that $\langle \bar{v}^{k+1}, v^{k+1}\rangle = \|v^k\|_\alpha^2$, it is easy then to show that $\|(1-\lambda)v^{k+1} + \lambda\bar{v}^{k+1}\|_\alpha^2 = \|v^k\|_\alpha^2 + O(\tau^3)$ too.

□

Appendix B: A family of implicit schemes

The following result prescribes a family of implicit integrators that also preserve the kinetic energy $\frac{1}{2}\langle v, m\rangle$.

If the operator D_α is self-adjoint w.r.t. to the inner product $\langle\cdot,\cdot\rangle$, and $\langle v, R(v,m)\rangle = 0$ for any v, m, then any scheme of the form:

$$D_\alpha \dot{v} = R(v^k + \frac{\tau}{2}\dot{v}, \hat{m}) \tag{10a}$$

$$v^{k+1} = v^k + \tau\dot{v} \tag{10b}$$

$$m^{k+1} = m^k + \tau D_\alpha \dot{v} \tag{10c}$$

preserves the kinetic energy $\frac{1}{2}\langle m, v\rangle$ exactly in the sense that $\langle m^k, v^k\rangle = \langle m^{k+1}, v^{k+1}\rangle$. This is independent of the choice of $\hat{m}$, which can be taken to be m^k (semi-explicit), m^{k+1} (implicit) or $m^{k+\frac{1}{2}} = m^k + \frac{\tau}{2}\dot{m}$ (mid-point).

Proof

$$\begin{aligned}\langle m^{k+1}, v^{k+1}\rangle - \langle m^k, v^k\rangle &= \langle D_\alpha v^{k+1}, v^{k+1}\rangle - \langle D_\alpha v^k, v^k\rangle \\ &= \langle D_\alpha(v^{k+1} - v^k), v^{k+1} + v^k\rangle \\ &= 2\langle \tau D_\alpha \dot{v}, \frac{v^{k+1} + v^k}{2}\rangle \\ &= \langle R(v^{k+\frac{1}{2}}, \hat{m}), v^{k+\frac{1}{2}}\rangle = 0\end{aligned}$$

□

Furthermore, if $D_\alpha v^k = m^k$, then $D_\alpha v^{k+1} = m^{k+1}$, so these schemes preserve the constitutive relation, contrary to the explicit scheme (5). The price to pay for this is that we need to assemble and solve a *different* system at each time step. Given that $R(v,m)$ (both in the continuous and in the fully discrete case) is a *bilinear* form, we can define the linear operators (matrices in the discrete case) $L_v^* := R(v,\cdot)$ and $P_m := R(\cdot, m)$. In the *semi-discrete case*, where $\hat{m} = m^k$, we need to solve at each time step a linear system of the

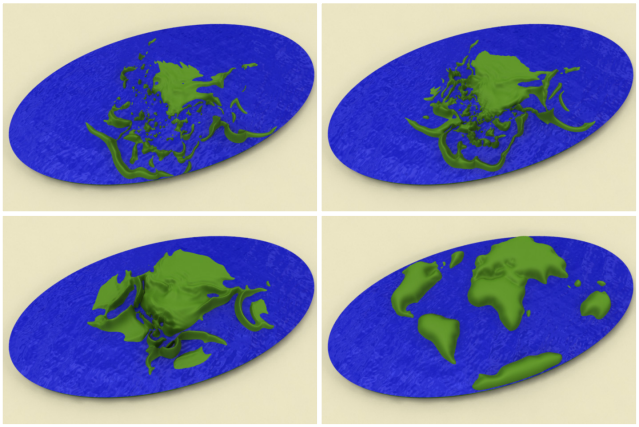

Figure 16: *Our SPI method is time invertible, allowing to generate seemingly random initial conditions (top, left) by flowing forward in time from the target setup (bottom, right) and then running the simulation with a negative time step (top, right and bottom, left).*

form

$$(D\alpha - \frac{\tau}{2}P_{m^k})v^{k+1} = (D\alpha + \frac{\tau}{2}P_{m^k})v^k \tag{11}$$

In the *mid-point (second order accurate) case*, where $\hat{m} = m^{k+\frac{1}{2}}$, we have to solve the non-linear system

$$f(\dot{v}) = D\alpha\dot{v} - R(v^k + \frac{\tau}{2}\dot{v}, m^k + \frac{\tau}{2}D\alpha\dot{v}) = 0 \tag{12}$$

Applying Newton's method to this system forces us to assemble and invert the Jacobian $Jf(\dot{v}) = D\alpha - \frac{\tau}{2}L^*_{v^k+\frac{\tau}{2}\dot{v}}D\alpha - \frac{\tau}{2}P_{m^k+\frac{\tau}{2}D\alpha\dot{v}}$ a couple of times per time step. This has proven in practice to be rather slow compared to the explicit integrator (5).