Geosci. Model Dev., 18, 6541–6551, 2025
https://doi.org/10.5194/gmd-18-6541-2025

Geoscientific Model Development
Open Access
EGU

Implementation of implicit filters for spatial spectra extraction

Kacper Nowak[1], **Sergey Danilov**[1], **Vasco Müller**[1], **and Caili Liu**[1,2]

[1]Alfred Wegener Institute, Helmholtz Center for Polar and Marine Research, Bremerhaven, Germany
[2]College of Oceanic and Atmospheric Sciences, Ocean University of China, Qingdao, China

Correspondence: Kacper Nowak (kacper.nowak@awi.de)

Received: 12 April 2024 – Discussion started: 25 July 2024
Revised: 27 May 2025 – Accepted: 23 June 2025 – Published: 29 September 2025

Abstract. Scale analysis based on coarse graining has been proposed recently as an alternative to Fourier analysis. It is now broadly used to analyze energy spectra and energy transfers in eddy-resolving ocean simulations. However, for data from unstructured-mesh models it requires interpolation to a regular grid. We present a high-performance Python implementation of an alternative coarse-graining method which relies on implicit filters using discrete Laplacians. This method can work on arbitrary (structured or unstructured) meshes and is applicable to the direct output of unstructured-mesh ocean circulation atmosphere models. The computation is split into two phases: preparation and solving. The first one is specific only to the mesh. This allows for auxiliary arrays that are then computed to be reused, significantly reducing the computation time. The second part consists of sparse matrix algebra and solving the linear system. Our implementation is accelerated by GPUs to achieve excellent performance and scalability. This results in processing data based on meshes with more than 10 million surface vertices in a matter of seconds. As an illustration, the method is applied to compute spatial spectra of ocean currents from high-resolution FESOM2 simulations.

1 Introduction

Atmospheric and oceanic motions span a wide range of spatial scales, each contributing differently to kinetic and available potential energy, energy generation, dissipation, and energy transfer across scales. Key questions include understanding how energy moves between scales, such as gyres, mesoscale and submesoscale motions in ocean dynamics. These processes are often described using the concept of an energy spectrum or cross-spectrum.

The most common technique to extract a spatial spectrum is the Fourier transform. However, when working with the output of ocean general circulation models (OGCMs), direct application of the Fourier transform is rarely possible as it requires data (samples) to be equally spaced as well as the domain to be in a rectangular shape (in global atmospheric configurations spherical harmonics are generally used). New convolution (coarsening)-based approaches to this problem have been proposed (Aluie et al., 2018; Sadek and Aluie, 2018; Aluie, 2019; Zhao and Aluie, 2025), and there already are multiple practical contributions showing the utility and significance of the proposed approach (see, for example, Schubert et al., 2020; Rai et al., 2021; Storer et al., 2023; Buzzicotti et al., 2023). While they solve some of the problems, like domain shape, they require data to be on a regular longitude–latitude grid. For vector fields in spherical geometry the procedure requires preliminary calculation of the Helmholtz decomposition.

Several recent OGCMs such as MPAS-Ocean (Ringler et al., 2013), FESOM2 (Danilov et al., 2017) and ICON-o (Korn, 2017) are based on either unstructured triangular meshes or their dual, quasi-hexagonal meshes. The use of the aforementioned coarse-graining method for the output of such models would require interpolation of the output data from native unstructured mesh to a regular mesh. This means additional computations. More importantly, the horizontal divergence of the interpolated velocities may show marked differences compared to the divergence on the original meshes.

These issues can be avoided if coarse graining is done on the original meshes. Recently a method has been proposed by Danilov et al. (2023) that solves this task. The method uses implicit filters based on discrete Laplacians. The discrete Laplacian operators can be constructed for arbitrary meshes and data placement on these meshes. This method

can therefore work on any mesh and can be applied directly to the output of unstructured-mesh models.

In this paper a high-performance Python implementation of the implicit filter method is presented, and practical examples of its usage are given. We use simulations performed with FESOM2 to illustrate the performance of the method. The discrete Laplacians depend on the mesh and data placement. For convenience, in Sect. 2 we recapitulate some mathematical details of the method and discretizations. The remaining part of this section discusses implementation. The results obtained with the implicit filters are compared with those produced by convolution based methods using velocity fields from simulation performed with FESOM2 on a global mesh with the resolution of 1 km in the Arctic Ocean in Sects. 3 and 4. The performance overview of the implementation is presented inSect. 5.

2 Implicit filter

2.1 Mathematical introduction

Let $\phi(\mathbf{x})$ be a scalar field, with $\mathbf{x}$ lying in some domain D. The goal is to find the distribution of the second moment of this field over spatial scales. This can be achieved using a coarse graining, akin to the methods presented by Aluie et al. (2018) and Sadek and Aluie (2018). However, coarse graining will rely on implicit filters, as proposed by Danilov et al. (2023). The coarsened field $\overline{\phi}_\ell(\mathbf{x})$ is found by solving

$$\left(1+\gamma\left(-\ell^2\Delta\right)^n\right)\overline{\phi}_\ell = \phi, \tag{1}$$

where Δ is the Laplacian, the smoothing scale is parameterized by ℓ and γ is a parameter that tunes the relation of $1/\ell$ to wavenumbers. Spectra can be computed using low-pass solutions as

$$\overline{E}_\ell = \frac{d}{k_\ell}\langle|\overline{\phi}_\ell|^2\rangle,$$

where the angular brackets denote spatial averaging and $k_\ell = 1/\ell$. An alternative method has been proposed recently by Zhao and Aluie (2025) which relies on high-pass filtering:

$$\overline{E}_\ell = -\frac{d}{k_\ell}\langle|\phi-\overline{\phi}_\ell|^2\rangle.$$

Below we will take $\gamma = 1/2$ for the low-pass method and $\gamma = 2$ for the high-pass method. As explained in Danilov et al. (2023), for the low-pass method $k_\ell = 1/\ell$ has the sense of wavenumber for $\gamma = 1/2$, and the wavelength is $\lambda = 2\pi\ell$. Such ℓ is related to the scale of box filter ℓ_{box} approximately as $\ell_{\text{box}}/\ell = 3.5$. The integer n defines the order of the implicit filter, as discussed in Guedot et al. (2015). For the high-pass method, analysis similar to that in Danilov et al. (2023) shows that the same interpretation of k_ℓ and ℓ is preserved if $\gamma = 2$. Such ℓ is related to the scale of box filter used with high-pass method approximately as $\ell_{\text{box}}/\ell = 6.1$.

The implicit coarsening procedure can be applied to a vector field $\boldsymbol{u}$ as

$$\left(1+\gamma\left(-\ell^2\boldsymbol{\Delta}\right)^n\right)\overline{\boldsymbol{u}}_\ell = \boldsymbol{u}. \tag{2}$$

In this case $\boldsymbol{\Delta}$ is the vector Laplacian, which includes metric terms in spherical geometry. In both cases of scalar and vector fields, Eqs. (1) and (2) are complemented by the boundary conditions of no normal flux (for $n = 1$) and additional conditions of no higher-order normal fluxes for $n > 1$.

The discrete Laplacian operator, used in this coarsening method, can be formulated for any computational mesh, whether it is structured or unstructured, or mesh geometry, whether it is flat or spherical. For unstructured meshes, Laplacians can be discretized through finite-volume or finite-element methods. Although mathematical details and discretizations were presented in Danilov et al. (2023), we briefly overview them here for convenience. Since discretizations of Laplacians on structured meshes are generally known, we focus below in this section on unstructured meshes.

Figure 1 presents schematics of several unstructured-grid discretizations in 2D view. In FESOM2, scalar degrees of freedom are placed at vertices, and median dual control volumes are used. They are obtained by connecting centroids of triangles with mid-edge points, as shown in panel Fig. 1a. The discrete horizontal velocities are placed on centroids of triangles (Fig. 1c). The placement of scalars in MPAS-Ocean differs by using the control volumes obtained by connecting circumcenters of triangles (Fig. 1b). These control volumes are Voronoi quasi-hexagonal polygons of the dual mesh (and vice versa, a triangular mesh can be considered the dual of the hexagonal one). The vector degrees of freedom are in this case the components of velocity normal to the edges of the scalar cells. In ICON-o, scalar degrees of freedom are placed at the circumcenters of triangles, and normal velocities are at mid-edges (Fig. 1d). The discretization of Laplacians depends on the placement of the degrees of freedom.

2.1.1 Scalar Laplacians

For median dual control volumes, we use the finite-element method, assuming first $n = 1$. The weak formulation of Eq. (1) is obtained by multiplying Eq. (1) by a sufficiently smooth function $w(\mathbf{x})$ and integrating over the domain D. This leads to

$$\int_D \left(w\overline{\phi}_\ell + \gamma\ell^2\nabla w\cdot\nabla\overline{\phi}_\ell\right)\mathrm{d}S = \int_D w\phi\,\mathrm{d}S.$$

The boundary term appearing after in the integration is zero by virtue of the boundary conditions. The discrete fields are expanded in series $\overline{\phi}_\ell = \sum_{v'}\overline{\phi}_{v'}N_{v'}(\mathbf{x})$ and $\phi = \sum_{v'}\phi_{v'}N_{v'}(\mathbf{x})$, where $N_v(\mathbf{x})$ is a P_1 piecewise linear basis function. This function equals 1 at the position of vertex v (see Fig. 1), decays to 0 at vertices connected to v by edges (like v' and

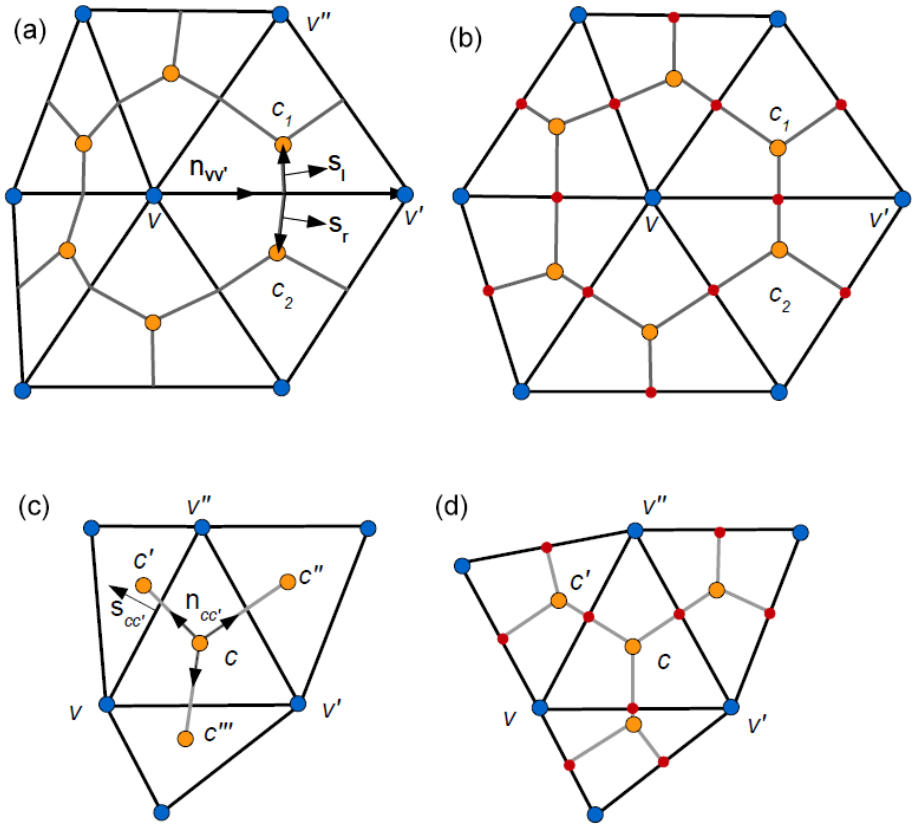

Figure 1. Schematics of several unstructured-grid discretizations: **(a)** median-dual control volumes around vertices are obtained by connecting centroids with mid-edge points (gray lines) and **(b)** a dual quasi-hexagonal mesh is obtained by connecting the circumcenters of triangles (on orthogonal meshes). The gray lines are perpendicular to edges, **(c)** centroids of triangles are used and **(d)** circumcenters of triangles are used.

v'') and is zero outside the stencil of nearest triangles. The continuous Galerkin approximation is obtained by requiring the weak equation above be valid for $w = N_v$ for any v. This results in

$$\left(\mathbf{M}_{vv'} + \gamma\tilde{\mathbf{D}}_{vv'}\right)\overline{\phi}_{v'} = \mathbf{M}_{vv'}\phi_{v'}.$$

Here, the summation over repeating v' is implied; $\mathbf{M}_{vv'} = \int N_v N_{v'} \mathrm{d}S$ is the mass matrix and $\tilde{\mathbf{D}}_{vv'} = \int \ell^2 \nabla N_v \cdot \nabla N_{v'} \mathrm{d}S$. Keeping the full mass matrix in this case does not improve the accuracy, and it can be replaced by its diagonally lumped approximation $\mathbf{M}^L_{vv'} = A_\mathrm{v}\mathbf{I}_{vv'}$, where A_v is the area of control volume associated with v. Note that the system matrix $\mathbf{S} = \mathbf{M}^L + (1/2)\tilde{\mathbf{D}}$ is symmetric and positive definite.

In the biharmonic case, $\mathbf{S} = \mathbf{M}^L + \gamma\tilde{\mathbf{D}}(\mathbf{M}^L)^{-1}\tilde{\mathbf{D}}$. The derivation procedure consists of two steps. One writes $\Delta\Delta\overline{\phi}_\ell = \Delta\overline{\psi}_\ell$, with $\overline{\psi}_\ell = \Delta\overline{\phi}_\ell$. Using the weak form of the last equality and expanding $\overline{\psi}_\ell$ in the same set of basis functions, one gets $\mathbf{M}^L_{vv'}\overline{\psi}_{v'} = \tilde{\mathbf{D}}_{vv'}\overline{\phi}_{v'}$. In the second step one applies the finite-element method to $\overline{\phi}_\ell + \gamma\Delta\overline{\psi}_\ell = \phi$. The flux terms that would appear in the weak equations are omitted by virtue of boundary conditions. The procedure can be generalized to higher-order filters.

The procedure above can be given a finite-volume treatment. Turning to Fig. 1a, we first compute $\nabla\overline{\phi}_\ell$ on triangles using three vertex values (v, v' and v'' for triangle c_1) and then combine fluxes through the segments of boundary (for edge vv' there are two segments with area vectors $\mathbf{s}_l$ and $\mathbf{s}_r$ taken with gradients at c_1 and c_2 respectively). Such treatment will lead to the same result.

For the quasi-hexagonal control volumes the $-\ell^2\Delta$ operator is expressed in a finite-volume way as

$$A_\mathrm{v}\left(\mathbf{D}\overline{\phi}_\ell\right)v = \ell^2 \sum_{v'\in N(v)} \frac{\overline{\phi}_v - \overline{\phi}_{v'}}{d_{vv'}} l_{vv'}, \tag{3}$$

where $d_{vv'}$ is the length of edge vv', and $l_{vv'}$ is the distance between circumcenters c_1 and c_2 on both sides of edge vv' (see Fig. 1b). The system matrix is

$$\mathbf{S}_{vv'}\overline{\phi}_{v'} = A_\mathrm{v}\phi_v, \quad \mathbf{S}_{vv'} = A_\mathrm{v}\mathbf{I}_{vv'} + \gamma A_\mathrm{v}\mathbf{D}_{vv'}, \tag{4}$$

with $\mathbf{I}_{vv'}$ being the identity matrix. The summation over repeating index v' is implied. On uniform meshes, $(\mathbf{M}^L)^{-1}\tilde{\mathbf{D}} = \mathbf{D}$. The matrix of the biharmonic operator can be obtained by applying the procedure used for $\mathbf{D}$ twice,

$$\mathbf{S}_{vv''} = A_\mathrm{v}\mathbf{I}_{vv''} + \gamma A_\mathrm{v}\mathbf{D}_{vv'}\mathbf{D}_{v'v''}. \tag{5}$$

For scalars at triangle circumcenters, the expression is similar to Eq. (3)

$$A_c\left(\mathbf{D}\overline{\phi}_\ell\right)c = \ell^2 \sum_{c'\in N(c)} \frac{\overline{\phi}_c - \overline{\phi}_{c'}}{d_{cc'}} l_{cc'}, \tag{6}$$

where A_c is the area of triangle c, $N(c)$ is the set of neighboring triangles, $l_{vv'}$ is the length of edge vv' and $d_{vv'}$ is the distance between the circumcenters of triangles with common edge vv' (see Fig. 1d). The biharmonic case is similar to the previous case. In the last two cases the expressions for Laplacians are simplified because of the orthogonality of the lines connecting centroids to edges.

2.1.2 Vector Laplacians

For the discretization of FESOM2 the easiest method is to seek for $\overline{\boldsymbol{u}}_\ell$ at vertices even though the discrete $\boldsymbol{u}$ values are at centroids. One reason is that the number of vertices is half as small, leading to a matrix problem of smaller dimensions. Due to the appearance of metric terms, equations for both components of $\overline{\boldsymbol{u}}_\ell$ are coupled, as explained in Danilov et al. (2023). The resulting matrix problem is $\mathbf{S}_2 = \mathbf{M}_2 + \gamma\mathbf{D}_2$ for the Laplacian filter and $\mathbf{S}_2 = \mathbf{M}_2 + \gamma\mathbf{D}_2(\mathbf{M}_2^L)^{-1}\mathbf{D}_2$ for the biharmonic filter. It acts on the vector $(\{\overline{u}_v\}, \{\overline{v}_v\})^T$. Here

$$\mathbf{D}_2 = \begin{pmatrix} \tilde{\mathbf{D}} & \mathbf{R} \\ -\mathbf{R} & \tilde{\mathbf{D}} \end{pmatrix}, \ \mathbf{M}_2 = \begin{pmatrix} \mathbf{M} & 0 \\ 0 & \mathbf{M} \end{pmatrix}.$$

The entries of matrix $\mathbf{R}$ are computed as $\mathbf{R}_{vv'} = \ell^2 \int m(-N_{v'}\partial_x N_v + N_v \partial_x N_{v'})\mathrm{d}S$. It is the operator accounting for metric terms linear in the metric factor $m = R_\mathrm{e}^{-1}\tan\theta$, where R_e is Earth radius, which is taken constant on triangles, and θ is the latitude. Compared to the scalar case, the entries of $\tilde{\mathbf{D}}$ are also modified by the metric terms as $\tilde{\mathbf{D}}_{vv'} = \ell^2 \int (\nabla N_v \cdot \nabla N_{v'} + (R_\mathrm{e}^{-2} + m^2) N_v N_{v'})\mathrm{d}S$. Finally, the right-hand side is obtained by projecting from cell to vertices: $\boldsymbol{u}_v = \mathbf{R}_{vc}\boldsymbol{u}_c$, where $\mathbf{R}_{vc} = \int N_v M_c \mathrm{d}S$, and M_c is the indicator function on triangle c. Summation is implied over repeating indices in matrix–vector products.

There are several options to do the filtering keeping $\overline{\boldsymbol{u}}_\ell$ at cells. For the stencil of nearest triangles (see Fig. 1c) the lines connecting the centroids on general meshes are not perpendicular to edges. For this stencil, there is no universally valid discretization for Laplacian. We use a simplified expression instead of true $-\ell^2\Delta$,

$$A_c(\mathbf{D}\boldsymbol{u})_c = A\ell^2 \sum_{c'\in C(c)} (\boldsymbol{u}_c - \boldsymbol{u}_{c'}), \tag{7}$$

which works stable in practice. One gets a valid discretization for $-\ell^2\Delta$ taking $A = 1$ on uniform quadrilateral meshes and $A = \sqrt{3}$ on uniform triangular meshes composed of equilateral triangles. On a triangular mesh obtained by splitting regular squares it corresponds to $-\partial_{xx} - \partial_{yy} \pm \partial_{xy}$ for $A = 3/2$, with the sign dependent on the direction the quadrilateral cells are split. The appearance of mixed derivatives is caused by the low symmetry of the mesh (only to rotations by 180°). Such meshes will make scale analysis essentially anisotropic; however all operators on such meshes have a similar mesh imprint. The operator (Eq. 7) is symmetric and positive semidefinite. In spherical geometry, the unit zonal and meridional vectors are different at c and c' locations. The account for this difference leads to metric terms that include the derivatives of unit directional vectors.

Other options for cell-based filtering of velocities will rely on a much larger stencil. Based on triangles c', c'' and c''' (see Fig. 1c) one can estimate $\nabla\overline{\boldsymbol{u}}_\ell$ on triangle c. Combining such estimates on c and c', the gradient will be estimated on edge vv'' and similarly on other edges, and then the divergence of such gradients will be computed at c. On uniform meshes this will involve a stencil of 10 triangles. Yet another method is to use the vector invariant form $\Delta = \nabla\nabla\cdot - \mathtt{curl}\,\mathtt{curl}$. For centroidal velocities both the discrete divergence and vorticity are naturally computed on vertices, through the cycle over the boundary of median-dual control volumes. The operation of gradient and second curl will involve three vertex values and return the result to cells. In this case the stencil will occupy 13 triangles on uniform meshes. In can be shown that such Laplacians are not more accurate than the vertex-based Laplacian but will lead to more expensive matrix vector multiplications in the solver procedure. We therefore have not tried these Laplacians thus far.

For the C-grid type of discretization of MPAS-Ocean and ICON-o the vector invariant form of Laplacians presents the main interest. The locations of normal velocities are given by small red circles in Fig. 1b and d. The divergence will be computed at vertices for the hex-C grid and at triangles for triangular C-grid, and vice versa for relative vortices. On uniform meshes 11 normal velocities will be involved in computations. These operators have not been implemented yet, but this might be done in future.

2.2 Implementation

The computational process is divided into two phases: initialization and solution. The initialization stage is independent of the filtering scale and involves precomputing auxiliary arrays that are reused during the solution phase. This optimization strategy significantly reduces the overall computational time, particularly for large-scale problems. Python 3 was chosen as the primary programming language due to its widespread adoption in the Earth sciences community and its extensive ecosystem of libraries. The implementation is publicly available as open-source software on GitHub.

2.2.1 Initialization

The implementation of the implicit filter method involves precomputing several auxiliary arrays based on the mesh connectivity matrix. These arrays are independent of the filtering scale and can be reused for multiple filter applications.

The non-zero coefficients of the **D** operator are determined using Eq. (3) and can be efficiently computed using sparse matrix algebra. To further enhance the computational performance, JAX's just-in-time compilation and vectorization capabilities were employed, resulting in a speedup of approximately 100× compared to a pure NumPy implementation. The precomputed auxiliary arrays can be saved to disk to reduce the computational overhead for subsequent filter applications. This approach leverages NumPy's file I/O capabilities and promotes efficient reuse of precomputed data across multiple filter scales. The time required for precomputation is approximately several minutes. In the case of high-resolution data (see Sect. 3) it took 15 m 40 s.

2.2.2 Solving linear system

After computing the coefficients of the **D** operator, they are assembled into a sparse matrix using SciPy's implementation of compressed sparse column (CSC) matrices. The **S** operator is then calculated based on either Eqs. (4) or (5). Following the suggestion of Guedot et al. (2015), the product of **S** and ϕ is subtracted from ϕ to simplify the solution of the linear system (Eq. 1). The resulting modified ϕ is then used by the conjugate gradient solver along with **S**. A solver tolerance of 10^{-9} was used for convergence.

Several alternative solvers, including the generalized minimal residual method (GMRES), were tested, but the conjugate gradient method consistently exhibited the best performance. The use of a preconditioner (incomplete LU factorization) was also investigated, but it did not lead to significant performance improvements.

To harness the parallel processing capabilities offered by GPUs, the CuPy library was employed. CuPy provides an identical interface to SciPy and requires minimal modifications to the method implementation. This library is optimized for NVIDIA GPUs but also supports graphics cards from other vendors, such as AMD. By utilizing CuPy, the algorithm maintains independence from both the operating system and hardware vendors.

2.3 Convolution filter

To make a comparison with the explicit filter, it is necessary to implement the box filter method proposed by Aluie et al. (2018). The crucial aspect involves defining the convolution kernel. The following formula was used for the low-pass method:

$$G(\mathbf{x}) = A\left(1 - \tanh\left(c\left(|\mathbf{x}| - 1.75/k_\ell\right)/a\right)\right). \tag{8}$$

Here, A is a normalizing factor, ensuring that $\int G d\mathbf{x} = 1$; c is the filter sharpness factor set to 1.5; and a is a mesh resolution. The combination $1.75/k_\ell$ corresponds to $\ell_{\mathrm{box}}/2$. It is replaced by $3.05/k_\ell$ for the high-pass method.

To implement this method, JAX was used to create the kernel matrix, while SciPy (in the case of CPU) and CuPy (in the case of GPU) were employed to apply the matrix to the data.

3 Data

The model data used were generated by the Finite-Element/volumE Sea ice-Ocean Model version 2 (FESOM2) simulation. FESOM2 is a multi-resolution sea ice–ocean model that solves the ocean primitive equations on unstructured meshes (Danilov et al., 2017). The sea ice module, that is part of the model, is formulated on the same meshes as the ocean module. Configuration of the model has horizontal resolution of 1 km in the entire Arctic Ocean, which smoothly coarsens to 30 km in the rest of the global ocean. There are 70 z levels in the vertical direction, with 5 m spacing within the upper 100 m. ERA5 atmospheric reanalysis fields (Hersbach et al., 2020) were used to force the model. The model was initialized from the PHC3 climatology (Steele et al., 2001) and run for 11 years starting from 2010. The first 4 years was considered a spinup. The realizations of velocity field from the last 7 years (2014–2020) were used in this work.

In order to be able to directly use the box filter on this dataset it was interpolated to a regular rectangular grid with 0.01° resolution. Prior to interpolation, the coordinate transform was performed such that the Arctic Ocean corresponds to the equatorial region in new coordinates. Using this rotation minimizes the error caused by using a regular longitude/latitude grid, since it ensures that the grid cells are very close to squares. Linear interpolation was used. As this is a very costly process, the domain was restricted to areas north of 73° latitude on the original mesh. The final mesh had dimension of 3200 by 3200 cells, resulting in 10 240 000 cell mesh.

4 Results

Figure 2 illustrates the application of the implicit filter for high-pass filtering of the velocity field. The left panel shows a snapshot of the absolute value of velocity at 70 m depth. Together with eddies and jet flows it shows a region with smoothed velocities, presumably caused by the sea ice drift. The velocity field in the right panel is obtained by subtracting the velocity field coarse grained with the scale of 100 km, which only leaves the small scales. Using the implicit filter allows one to perform this operation on the native mesh and on the spherical Earth surface, without the need of regridding the data. The continental-break currents are rather strong and carry a significant part of kinetic energy. As is seen comparing the panels of Fig. 2, they contribute very substantially to a large-scale part of the flow. One can expect therefore that energy spectra computed for the entire Arctic Ocean will have an elevated spectral density at large scales.

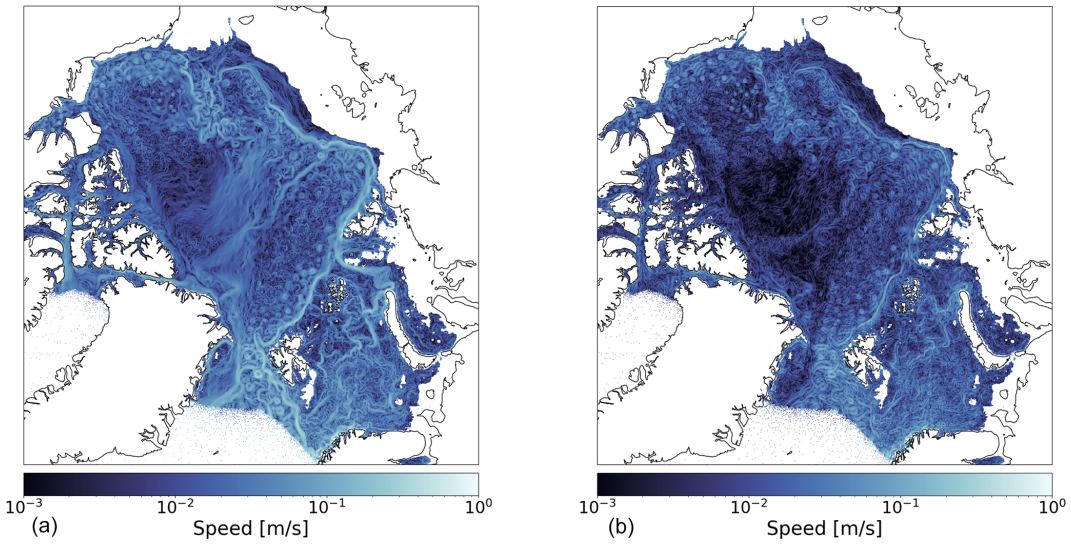

Figure 2. A snapshot of ocean currents at 70 m depth (logarithmic scale). Panel **(a)** shows the simulated results, while panel **(b)** shows the results of high-pass filtering with the scale of 100 km. The implicit Laplacian filter has been used over data on a native unstructured grid, taking into account spherical geometry.

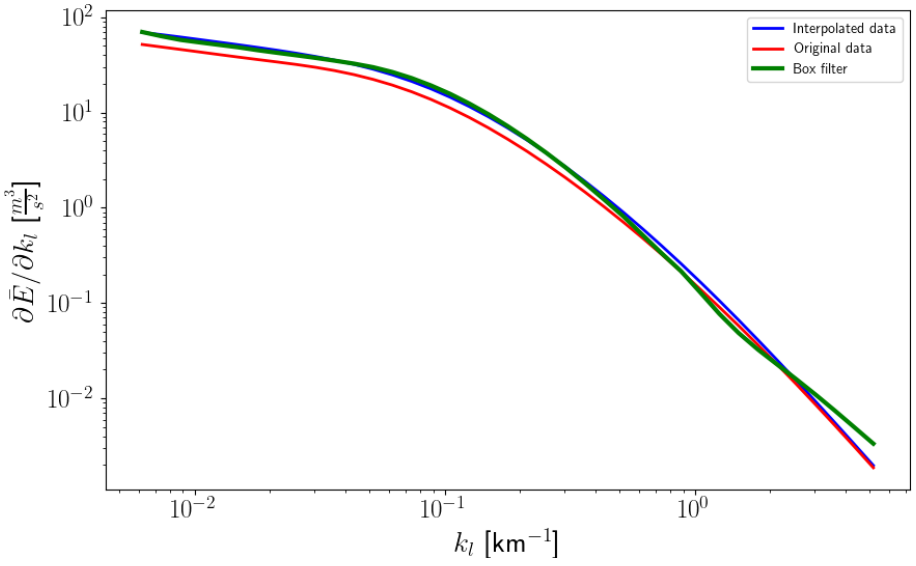

Figure 3. Wavenumber spectra for the entire Arctic Ocean. The red line corresponds to the implicit harmonic filter applied to the data on the original grid. The blue and green lines correspond to the implicit harmonic and explicit box filter applied to the interpolated data. The highest wavenumber is π/h, where h is the height of triangles of the original grid.

Figure 3 presents kinetic energy spectra obtained with the implicit Laplacian filter using the original data (on the original triangular mesh) (red line) and interpolated data (blue). They are compared with the spectrum obtained by coarse graining with the explicit box filter (green). The latter is computed ignoring the Earth curvature (the cosine of latitude is replaced with 1), enabling the convolution via the Fourier transform. The implicit filter allows us to compute the spectra of the interpolated data on both the longitude–latitude mesh and on its flat geometry approximation. Although the coarse-grained velocities show some small differences (25 % at the largest scale) in these two cases, the energy spectra turn out to be almost identical, which substantiates the approach taken for the box filter.

As seen in Fig. 3, the implicit Laplacian filter and the explicit box filter provide matching results (1 % difference at the largest scale). The largest wavenumber is $\pi/h \approx= 4$ cycle per kilometer, where h is the height of triangles of the original mesh. One can note that for larger scales there is a small shift between the results computed on original and reg-

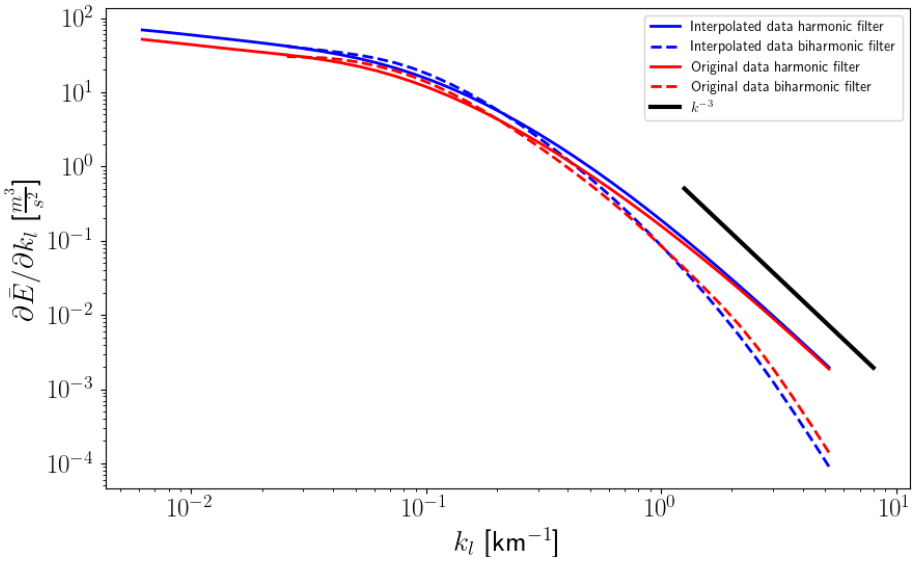

Figure 4. Comparison between the spectra computed with the box filter and the implicit harmonic filter applied to data on native grid and interpolated to regular grid.

ular meshes. We guess that this is related to the effect of no-flux boundary conditions (Danilov et al., 2023), which are applied along the boundary between water and land on the original mesh but along the boundary of interpolation mesh in the other case. Interestingly, for the interpolated data, the spectra for the implicit and box filter are very close despite the difference in boundary conditions. Note that the spectra obtained using the implicit Laplacian filter are smoother than spectra based on the box filter. This correlates with better spectral sensitivity of the latter, which is, however, worse than the sensitivity of the biharmonic filter.

In Fig. 4 we compare the spectra computed with harmonic and biharmonic filters. For the biharmonic filters, the conjugate gradient algorithm requires more iterations to achieve required tolerance. This results in longer computation, and on large scales it might not converge, so we stopped on the wavenumber $k_\ell \approx 0.025$ cycle per km, which corresponds to the wavelength of 250 km. While the spectra are close on the wavelength larger than approximately 12 km, they start to deviate at smaller wavelengths. This could have a numerical explanation at wavelengths that correspond to grid scales, as discrete Laplacians deviate from their continuous counterpart at grid scales, and this deviation is larger in biharmonic operators. However, the spectra in Fig. 4 also disagree at larger scales, which is an indication that the real spectra have a slope steeper than -3 at these scales. We give elementary illustrations in Sect. 6.

5 Performance benchmarks

To facilitate the computational demands of this study, extensive computational resources were generously provided by the Jülich Supercomputing Centre (JSC). As the primary computing environment, a single node from the JUWELS Booster Module was used. This node was made of two AMD EPYC Rome 7402 CPUs, 512 GB of DDR4 RAM and four NVIDIA A100 GPUs, each equipped with 40 GB of HBM2e memory. To optimize computational efficiency and resource utilization, only a single GPU was employed for the duration of this study.

The performance evaluation of the implicit filter method focuses on its computational efficiency, assessed by measuring the execution time required to process data. However, data access from disk can introduce substantial overhead, potentially influencing the overall execution time. To isolate the computational efficiency of the method, IO time, representing the duration spent reading and writing data to disk, is excluded from the execution time measurements. To ensure the accuracy and consistency of time measurements, each configuration is executed 10 times, and the mean value is taken.

As evident from Fig. 5, the performance of the implicit filter method exhibits a linear relationship with mesh size above 10^6 nodes. Furthermore, the method effectively handles meshes with over 11 million nodes, achieving processing times of approximately 5 s for a 100 km filter scale. Such remarkable efficiency is attributed to the method's inherent scalability, enabling it to process data efficiently on increasingly large meshes without performance degradation.

This capability to handle large meshes is essential for analyzing real-world datasets, which often cover vast areas and demand high-resolution meshes for accurate representation. The implicit filter's scalability ensures its effectiveness in processing these large datasets, making it suitable not only for current state-of-the-art meshes but also for future generations of increasingly high-resolution meshes.

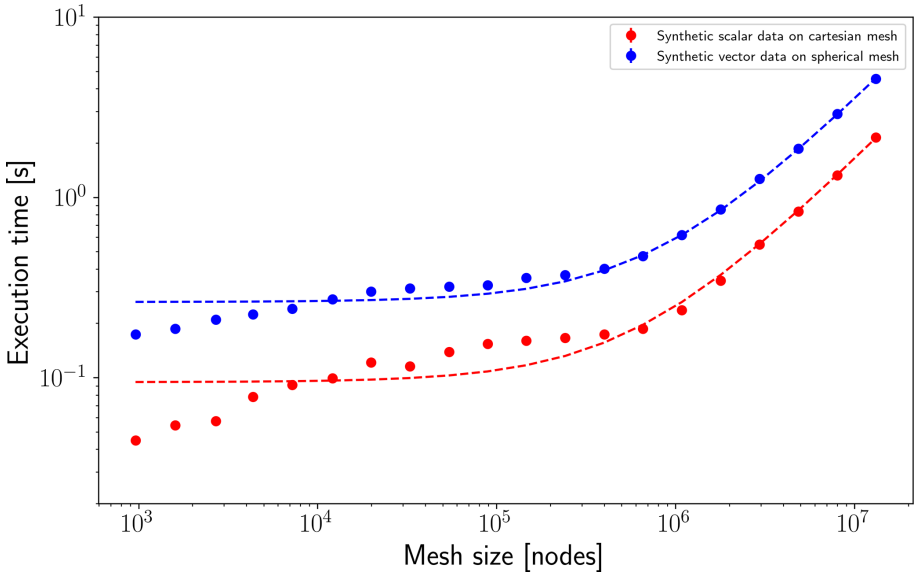

Figure 5. Execution time of implicit filter on synthetic data. Presenting data on both a Cartesian and a spherical mesh. Filter size of 100 km was used at all cases. Dashed lines illustrate fitted linear functions.

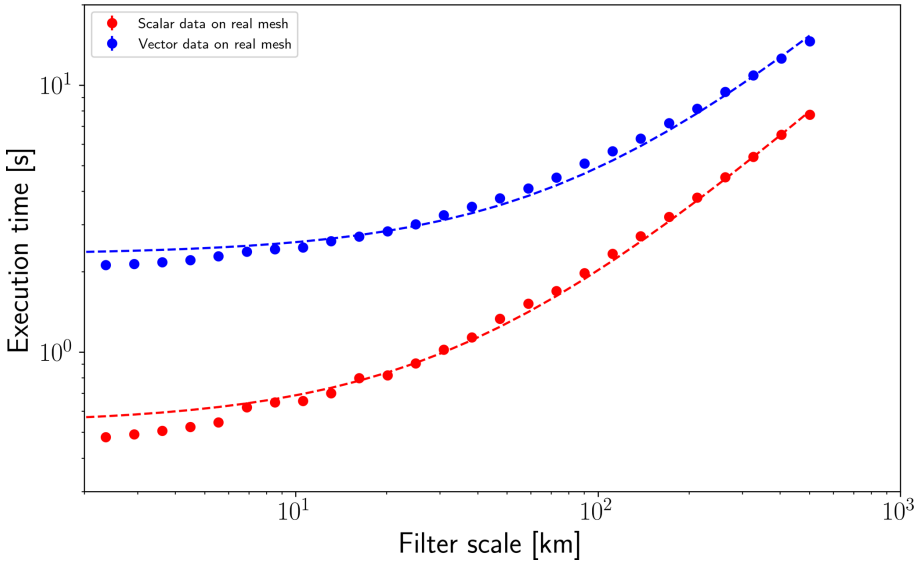

Figure 6. Execution time of implicit filter on FESOM2 output. Results show computation time of both vector and scalar data on a mesh with 11 million nodes (see Sect. 3). Dashed lines illustrate fitted linear functions.

For the benchmarks, a fully unstructured mesh was used, which has 11 538 465 surface nodes. The measured results of the execution time, as shown in Fig. 6, exhibit a close-to-linear dependency with filter scale for those larger than 50 km. In particular, computation for a filter scale up to approximately 100 km is done within 6 s, even for a mesh with more than 10 million nodes and a resolution of about 1 km in the focus area. As this is the range of scales that is of the most interest, it shows remarkable performance. Convergence becomes more challenging for larger scales, and results diverge from a linear dependency. The scales where those issues arise depend on specific data, but the scale is typically larger than 1000 km.

6 Some comments

The present implementation supports triangular meshes of FESOM2 and produces coarse-grained fields at mesh vertices. Since the original discrete horizontal velocities in FESOM2 are at the centers of triangles, computation of coarse-grained velocities at triangles was also attempted and found

to lead to very close results regarding the kinetic energy spectra. Vertex computations should therefore be preferred, as they require smaller matrices. Nevertheless, triangle-based computations might be required if dissipation is studied, as the dissipation tendency is noisy. They are under implementation and are based on the simplified Laplacian (Eq. 7). The support of C-grid type discretization for ICON-o discretization (scalar Laplacians for data on circumcenters of triangles and vector-invariant Laplacians for velocities normal to edges) and the support of regular quadrilateral C-grids will be added in the near future. Our aim is a tool that supports different meshes.

The calculation of spectra can be based on the low-pass method, as proposed by Sadek and Aluie (2018) and used in Danilov et al. (2023). A new method, based on high-pass filtering, was proposed recently by Zhao and Aluie (2025), after this paper was submitted. The method only modifies the calculations of spectra but relies on the same coarse-grained fields as the low-pass method. We added it to our implementation, which required minor updates. According to the analysis by Zhao and Aluie (2025), the new method can handle much steeper spectra than the low-pass method, and we illustrate some points here. As discussed in Danilov et al. (2023), the low-pass method based on implicit Laplacian filter can handle spectra which are not steeper than k^{-3}. The use of implicit biharmonic filter extends this range to k^{-5}. Based on the analysis of Zhao and Aluie (2025), these ranges extend to k^{-5} and k^{-7} respectively with the high-pass method.

In Fig. 7 we present the spectra computed using the implicit filters of different order given the Fourier spectrum $E_k \sim k^{-2}/(1+(k/k^*)^2)$, with $k^* = 30\,k_{\min}$, where $k_{\min}$ corresponds to the domain wavelength taken as 2048 km. The spectra were computed using the analytical expression for the form factor and the Fourier symbol of one-dimensional discrete Laplacian given in Danilov et al. (2023). The dark- and light-blue (thick black and gray) curves correspond, respectively, to the Laplacian and biharmonic filters used with the low-pass (high-pass) method.

While one expects that the Laplacian filter will fail for large k in the case of low-pass method, the dark-blue line has a slope flatter than -3 over a range of wavenumbers where the real slope is -4. One can erroneously interpret this interval as a true spectrum (since it is flatter than -3). This behavior is caused by aliasing from the side of small wavenumbers where the spectral energy density is much larger. With the high-pass method the spectrum is much closer to the Fourier spectrum even for the Laplacian filter (thick black line) and is only slightly worse than the result with biharmonic filter for the low-pass method. Increasing the order of filter leads to very accurate behavior (gray curve). Thus, while the high-pass method should be a preferred one, there are also arguments in favor of higher-order filters.

The illustration in Fig. 7 stresses the fact that one generally needs to test the results obtained by sequential filtering if they show spectral slopes approaching their critical limits. In any

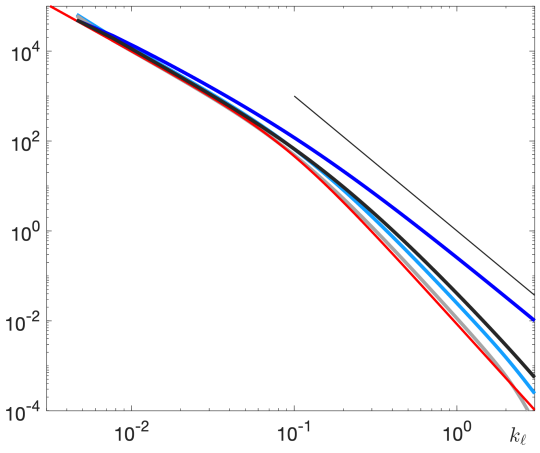

Figure 7. Theoretical energy spectra obtained using Laplacian (dark blue) and bi-Laplacian (light blue) implicit filters with low-pass method. The thick black and gray curves show the same but for the high-pass method. The Fourier spectrum is shown by the thin red line, and the straight black line corresponds to a slope of -3.

case, as mentioned by Zhao and Aluie (2025), this is a strong argument in favor of the high-pass method or higher-order filters. Guided by this fact, the behavior shown in Fig. 4, and the possibility to compute the Fourier spectrum for the interpolated data, we compare the spectra obtained with implicit harmonic and biharmonic filters with the Fourier spectrum in Fig. 8. The spectrum obtained with the harmonic filter deviates from the Fourier spectrum, but the biharmonic filter follows the Fourier spectrum very closely. This issue with the harmonic filter can be eased using the high-pass filter method, resulting in spectra with a slope steeper than -3, without the need for the additional compute required by the biharmonic filter.

The use of the biharmonic filter is computationally more expensive, and even worse, the convergence can be lost for large ℓ, which correspond to wavelengths of domain size in the case of very fine meshes if the conjugate gradient solver is used. At present, we rely on the conjugate gradient solver available in Python, and we work on preconditioners and solution methods that will remove these difficulties. The improved convergence for biharmonic filter is important, as it opens up perspectives of using filters of higher order, as explained in Guedot et al. (2015) and Danilov et al. (2023).

It is reminded that the wavenumber scale $k_\ell = 1/\ell$ used by us corresponds to the wavenumber of the Fourier spectrum. This means that ℓ corresponds to $\lambda/(2\pi)$, with λ the wavelength. This requires care when comparing our approach to that based on other filters, as their meaning of ℓ might be different. For the box-type filter, $\ell_{\text{box}} \approx 3.5\ell$ or 6.1ℓ for the low-pass and high-pass method.

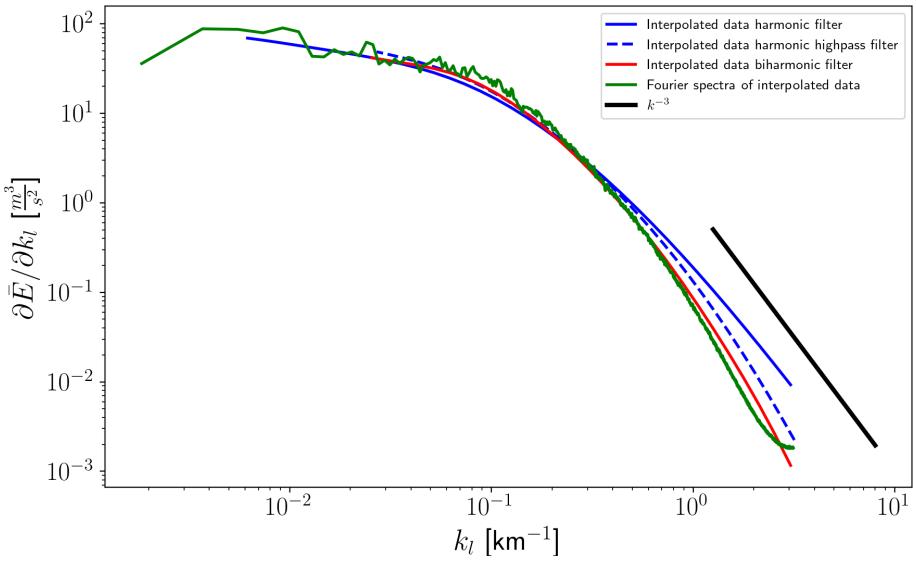

Figure 8. Fourier energy spectrum compared to the spectra obtained with implicit harmonic and biharmonic filters for the data interpolated to a regular grid. For the harmonic filter, both low-pass and high-pass methods were used.

7 Conclusions

This work presents a high-performance implementation of a novel method for extracting spatial spectra from unstructured mesh data, offering a compelling alternative to conventional methods. The open-access code and elementary documentation can be found at GitHub (https://github.com/FESOM/implicit_filter, last access: 25 August 2025) and Zenodo (Nowak and Danilov, 2024). Unlike its predecessors, the implicit filter method directly operates on unstructured meshes, such as triangular and quasi-hexagonal meshes, eliminating the need for computationally expensive interpolation to regular grids. This capability makes the implicit filter directly applicable to the output of unstructured-mesh ocean circulation models, surpassing the limitations of traditional methods.

To enhance practical applicability, the implicit filter method is implemented in Python using a high-performance algorithm that employs a two-phase approach to optimize computational efficiency. The first phase involves precomputing mesh-specific data, significantly reducing the computational load during the actual filtering process. This optimization strategy ensures efficient resource utilization and minimizes overall execution time. The second phase can use GPU-accelerated sparse matrix algebra. Depending on the conditions, use of the GPU reduces computation time by up to 2 orders of magnitude. This computational prowess enables the processing of high-resolution data from meshes with millions of surface vertices within seconds.

The efficacy of the implicit filter method is demonstrated by applying it to compute spatial spectra of ocean currents from high-resolution general circulation model output. The results obtained from the proposed method exhibit agreement with those obtained using traditional methods, such as a box filter, validating its accuracy and robustness. Furthermore, the method's ability to handle unstructured meshes directly provides a more comprehensive analysis compared to traditional methods that require interpolation to a regular grid. However one needs to note that for spectra with slopes steeper than -3, a biharmonic filter needs to be used.

Code and data availability. Source code, along with documentation and examples, is available at https://github.com/FESOM/implicit_filter and Zenodo (https://doi.org/10.5281/zenodo.10907365, Nowak and Danilov, 2024). All scripts used for making figures presented in this work are available on Zenodo (https://doi.org/10.5281/zenodo.10957614, Nowak et al., 2024).

Data used for Fig. 3 are not included as they would require files with size exceeding the capacity of the Zenodo repository. As this figure is only for illustrative purposes, a similar figure can be obtained using any data and included example scripts (https://doi.org/10.5281/zenodo.10907365, Nowak and Danilov, 2024).

Author contributions. KN: software development, manuscript writing, data analysis, SD: manuscript writing, supervision, VM: running simulation, software testing, CL: processing data, software development

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

Disclaimer. This publication is part of the EERIE project funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Climate Infrastructure and Environment Executive Agency (CINEA). Neither the European Union nor the granting authority can be held responsible for them.

Publisher's note: Copernicus Publications remains neutral with regard to jurisdictional claims made in the text, published maps, institutional affiliations, or any other geographical representation in this paper. While Copernicus Publications makes every effort to include appropriate place names, the final responsibility lies with the authors.

Acknowledgements. This work has received funding from the Swiss State Secretariat for Education, Research and Innovation (SERI) under contract 22.00366. This work was funded by UK Research and Innovation (UKRI) under the UK government's Horizon Europe funding guarantee (grant number 10057890, 10049639, 10040510, 10040984). It was also funded by the European Commission, HORIZON EUROPE Framework Programme (grant no. 101081383).

This work is also a contribution to projects M3 and S2 of the Collaborative Research Centre TRR181 "Energy Transfer in Atmosphere and Ocean", funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – project no. 274762653.

Authors acknowledge usage of ChatGPT by OpenAI version 3.5 in preparation of the manuscript. It was used for linguistic and editorial purposes in an earlier draft.

Financial support. This research has been supported by the European Commission, HORIZON EUROPE Framework Programme (grant no. 101081383); the UK Research and Innovation (grant nos. 10057890, 10049639, 10040510 and 10040984); the Staatssekretariat für Bildung, Forschung und Innovation (grant no. 22.00366); and the Deutsche Forschungsgemeinschaft (grant no. 274762653).

The article processing charges for this open-access publication were covered by the Alfred-Wegener-Institut Helmholtz-Zentrum für Polar- und Meeresforschung.

Review statement. This paper was edited by Olivier Marti and reviewed by Ian Grooms and one anonymous referee.

References

Aluie, H.: Convolutions on the sphere: Commutation with differential operators, GEM-Int. J. Geomath., 10, 9, https://doi.org/10.1007/s13137-019-0123-9, 2019.

Aluie, H., Hecht, M., and Vallis, G. K.: Mapping the energy cascade in the North Atlantic Ocean: The coarse-graining approach, J. Phys. Oceanogr., 48, 225–244, https://doi.org/10.1175/jpo-d-17-0100.1, 2018.

Buzzicotti, M., Storer, B. A., Khatri, H., Griffies, S. M., and Aluie, H.: Spatio-temporal coarse-graining decomposition of the global ocean geostrophic kinetic energy, J. Adv. Model. Earth Syst., 15, e2023MS003693, https://doi.org/10.1029/2023MS003693, 2023.

Danilov, S., Sidorenko, D., Wang, Q., and Jung, T.: The Finite-volumE Sea ice–Ocean Model (FESOM2), Geosci. Model Dev., 10, 765–789, https://doi.org/10.5194/gmd-10-765-2017, 2017.

Danilov, S., Juricke, S., Nowak, K., Sidorenko, D., and Wang, Q.: Computing spatial spectra using implicit filters, Zenodo [data set], https://doi.org/10.5281/zenodo.8171835, 2023.

Guedot, L., Lartigue, G., and Moureau, V.: Design of implicit high-order filters on unstructured grids for the identification of large-scale features in large-eddy simulation and application to a swirl burner, Phys. Fluids, 27, 045107, https://doi.org/10.1063/1.4917280, 2015.

Hersbach, H., Bell, B., Berrisford, P., Hirahara, S., Horányi, A., Muñoz Sabater, J., Nicolas, J., Peubey, C., Radu, R., Schepers, D., Simmons, A., Soci, C., Abdalla, S., Abellan, X., Balsamo, G., Bechtold, P., Biavati, G., Bidlot, J., Bonavita, M., and Thépaut, J.-N.: The ERA5 global reanalysis, Q. J. Roy. Meteorol. Soc., https://doi.org/10.1002/qj.3803, 2020.

Korn, P.: Formulation of an unstructured grid model for global ocean dynamics, J. Comput. Phys., 339, 525–552, 2017.

Nowak, K. and Danilov, S.: Implicit filter: v1.0.0, Zenodo [data set and code], https://doi.org/10.5281/zenodo.10907365, 2024 (code also available at: https://github.com/FESOM/implicit_filter, last access: 25 August 2025).

Nowak, K., Danilov, S., Müller, V., and Liu, C.: Implementation of implicit filter for spatial spectra extraction, Zenodo [data set], https://doi.org/10.5281/zenodo.10957614, 2024.

Rai, S., Hecht, M., Maltrud, M., and Aluie, H.: Scale of oceanic eddy killing by wind from global satellite observations, Sci. Adv., 7, eabf4920, https://doi.org/10.1126/sciadv.abf4920, 2021.

Ringler, T., Petersen, M., Higdon, R., Jacobsen, D., Maltrud, M., and Jones, P.: A multi-resolution approach to global ocean modelling, Ocean Model., 69, 211–232, 2013.

Sadek, M. and Aluie, H.: Extracting the spectrum of a flow by spatial filtering, Phys. Rev. Fluids, 3, 124610, https://doi.org/10.1103/physrevfluids.3.124610, 2018.

Schubert, R., Gula, J., Greatbatch, R. J., Baschek, B., and Biastoch, A.: The Submesoscale Kinetic Energy Cascade: Mesoscale Absorption of Submesoscale Mixed Layer Eddies and Frontal Downscale Fluxes, J. Phys. Oceanogr., 50, 2573–2589, https://doi.org/10.1175/JPO-D-19-0311.1, 2020.

Steele, M., Morley, R., and Ermold, W.: PHC: A Global Ocean Hydrography with a High-Quality Arctic Ocean, J. Climate, 14, 2079–2087, https://doi.org/10.1175/1520-0442(2001)014<2079:PAGOHW>2.0.CO;2, 2001.

Storer, B. A., Buzzicotti, M., Khatri, H., Griffies, S. M., and Aluie, H.: Global cascade of kinetic energy in the ocean and the atmospheric imprint, Sci. Adv., 9, eadi7420, https://doi.org/10.1126/sciadv.adi7420, 2023.

Zhao, D. and Aluie, H.: Calculating spectra by sequential filtering, J. Renew. Sustain. Energ., 17, 013303, https://doi.org/10.1063/5.0244909, 2025.