Momentum Budget Evaluation in ASTE Release 1
Part I: Full momentum budget

Helen Pillar *[1], An T. Nguyen[1], Jean-Michel Campin[2] and Patrick Heimbach[1]

[1]Oden Institute for Computational Engineering and Sciences, UT Austin, TX
[2]Department of Earth, Atmospheric and Planetary Sciences, MIT, MA

June 14, 2021

1 Introduction

The purpose of these notes is to describe how to perform accurate momentum budget analyses using output from the first release of the Arctic and Subpolar gyre sTate Estimate [ASTE_R1 Nguyen et al. 2021b]. The goal of these analyses is to partition, at the grid-point level, the rate of change of momentum into all of its contributing terms in the momentum equation, such as wind and Coriolis forces, horizontal advection, resolved diffusion of momentum, parameterized diffusion of various kinds, etc. We refer to "closing the budget" when the sum of all terms in the momentum equation accurately balance the total Eulerian tendency.

ASTE_R1 has been produced using the non-linear inverse modeling framework developed within the consortium for Estimating the Circulation and Climate of the Ocean [ECCO, Stammer et al. 2002, Wunsch and Heimbach 2007, Forget et al. 2015, Heimbach et al. 2019]. The inversion consists of an iterative, gradient-based minimization of a least-squares model-data misfit cost-function. For our purposes, it is important to highlight here that misfit minimization is achieved by adjusting only uncertain independent variables that serve as model inputs (also referred to as "controls"), comprising initial conditions, atmospheric state variables and ocean mixing parameters. This choice ensures exact adherence to the conservation laws encapsulated in the model, permitting meaningful budget analyses, including for momentum and vorticity. This is a key advantage of ECCO-based products over model-data reanalyses produced using sequential data assimilation, for which analysis increments can introduce spurious sources and sinks of basic properties [Wunsch and Heimbach 2007, Stammer et al. 2016].

ASTE_R1 provides a data-constrained and dynamically-consistent estimate of the ocean and sea-ice states for the period 2002-2017. The nominal horizontal resolution in ASTE, based on the LLC-270 grid [Forget et al. 2015], is 14 km in the Arctic. A full description of ASTE_R1 production and assessment of the solution – including extensive comparison to available observations – is presented by Nguyen et al. [2021b]. Additional user notes, including information on data distribution and post-processing tools, are given in Nguyen et al. [2021a]. The reader is also referred to

*helen.pillar@utexas.edu

Piecuch [2017] for very helpful guidance in evaluating volume, heat and salt budgets.

In Part 1 of the momentum budget notes presented here, modeled momentum tendencies and necessary diagnostic output required to close the full momentum budget in ASTE_R1 are described. In Part 2 of these notes, offline diagnosis of the rotational momentum budget will be described to support use of the `ROTMOM` matlab toolbox (in development). The dynamical core of all ECCO-based model-data syntheses is the Massachusetts Institute of Technology general circulation model [MITgcm, Marshall et al. 1997b,a] and we highlight that the model manual [Adcroft et al. 2018, regularly updated at https://mitgcm.readthedocs.io/en/latest/] is a wealth of information containing detailed description of many of the routines referenced below. Note that here only the vector invariant form of the momentum equation (as used in ASTE_R1) is considered. The manual provides some very useful direction for diagnosing the full momentum budget for both the vector invariant and the flux-form momentum equations; many of our notes on the discretization have been taken from here and checked against the ASTE_R1 code. We are grateful to the larger MITgcm developer community for also posting useful discussion on the MITgcm support list and in subroutine annotations. Finally, we note that though our discussion focuses on momentum budget evaluation using the ASTE_R1 configuration, based on checkpoint 65q of the model, we have tried to make our notes helpful for all MITgcm users (i.e., using other ECCO products or MITgcm checkpoints).

2 Governing Momentum Equation

The general form of the governing Navier Stokes momentum equation may be written as:

$$\frac{\partial \mathbf{u}}{\partial t} = \Sigma\, \mathbf{G} - \frac{1}{\rho_0}\boldsymbol{\nabla} p, \tag{1}$$

with $\boldsymbol{\nabla} \cdot \mathbf{u} = 0$ and $\mathbf{u} = \mathbf{u_b}$ on the boundary. Here consideration is restricted to domains with impermeable boundaries so that $\hat{\mathbf{n}} \cdot \mathbf{u_b} = 0$, where $\hat{\mathbf{n}}$ is a unit vector normal to the boundary. Here $\mathbf{u} = [u, v, w]$ is the 3-dimensional velocity field, t is time, p is the pressure, and ρ_0 is the reference density. $\mathbf{G} = [G^x, G^y, G^z]$ is the contribution from any single momentum source, sink or redistribution term, including inertia, all body forces and the deviatoric component of the stress tensor (i.e., internal viscosity). For incompressible fluids, it is useful to retain the volumetric stress tensor, ∇p, outside of $\mathbf{G}$ for the following reason. Via the Helmholtz decomposition, all terms contained within $\mathbf{G}$ can be split into purely divergent and purely rotational components. It is helpful to inspect the latter in isolation as they project onto the local acceleration of the non-divergent flow, $\partial \mathbf{u}/\partial t$, whilst the divergent components project only on to the pressure gradient, ∇p. This separation is also exploited in the numerical solution of the nonlinear flow [Marshall et al. 1997a]. Taking the divergence of Eq. (1) eliminates the local acceleration, $\partial \mathbf{u}/\partial t$, yielding a Poisson equation which we invert for the pressure, p. This step requires knowledge of the source terms $\mathbf{G}$, which in turn requires knowledge of the non-divergent velocity, $\mathbf{u}$, available only for past time steps. The pressure, p, is substituted into Eq. (1) to obtain a first guess of the velocity, $\mathbf{u}$. This estimate is then adjusted by computing and applying the pressure correction (i.e., volumetric stress perturbation) necessary to ensure non-divergence before proceeding to the next time step.

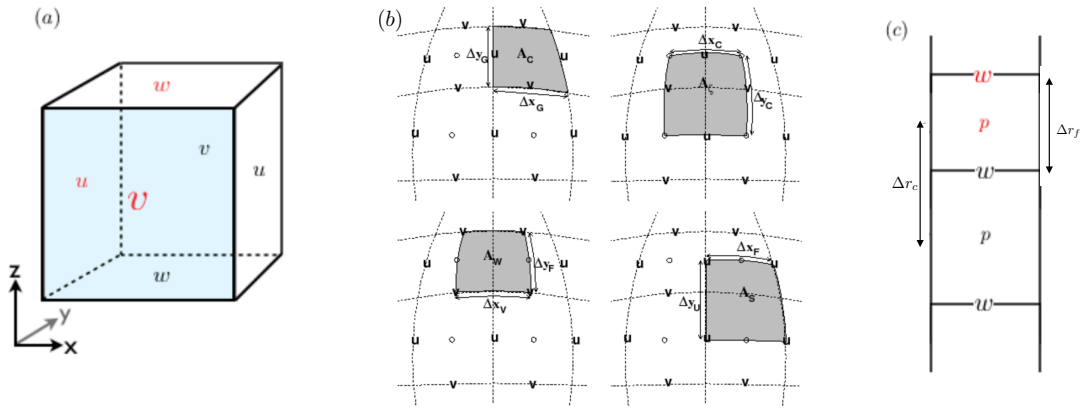

Figure 1: C-grid staggering of velocity components, $[u, v, w]$, and pressure, p, in the MITgcm; the (i, j, k) point is shown in red in (a) and (c). Convention is the i and j indices increase in the positive x and y directions, respectively. The k index increases in the negative z direction. Horizontal grid descriptors (lengths and areas) are shown in panel (b), copied from the MITgcm manual [Adcroft et al. 2018]. Vertical spacings are shown in (c). Partially filled cells are enabled in the MITgcm, offering a finer representation of bathymetric features. The thickness of a cell centred on a $[u, v, p, \zeta]$-point is given by $\Delta r_f \cdot h_{[w,s,c,\zeta]}$, respectively, where $h \in [0, 1]$ is the open fraction.

Expanding $\mathbf{G}$ into separate contributions from Coriolis, inertia, buoyancy, wind, and viscous stresses:

$$\frac{\partial \mathbf{u}}{\partial t} = -\,2\boldsymbol{\Omega} \times \mathbf{u} \;-\; \mathbf{u} \cdot \boldsymbol{\nabla}\mathbf{u} \;+\; \mathbf{b} \;+\; \frac{1}{\rho_0}\boldsymbol{\nabla} \cdot \tau \;-\; \frac{1}{\rho_0}\boldsymbol{\nabla} p, \tag{2}$$

where $\boldsymbol{\Omega}$ is the Earth's rotational vector, $\mathbf{b} = -\mathbf{g}\rho'/\rho_0\hat{\mathbf{k}}$ is the buoyancy force, and $\boldsymbol{\tau}$ includes applied surface wind and internal viscous stresses (e.g., see Griffies & Adcroft 2008).

Using vector identities, inertia in Eq. (2) can be re-written as follows:

$$-\,\mathbf{u} \cdot \boldsymbol{\nabla}\mathbf{u} = -\boldsymbol{\zeta} \times \mathbf{u} \;-\; \boldsymbol{\nabla}\left[\frac{1}{2}(\mathbf{u} \cdot \mathbf{u})\right] \;=\; -\boldsymbol{\zeta} \times \mathbf{u} \;-\; \boldsymbol{\nabla}\, KE, \tag{3}$$

where $\boldsymbol{\zeta} = \boldsymbol{\nabla} \times \mathbf{u}$ is the three-dimensional vorticity and $KE = 1/2(\mathbf{u} \cdot \mathbf{u})$ is the kinetic energy per unit mass of a fluid parcel. The full vector-invariant momentum equation is then given by:

$$\frac{\partial \mathbf{u}}{\partial t} = \;-\,(2\boldsymbol{\Omega} \times \mathbf{u}) \;-\; (\boldsymbol{\zeta} \times \mathbf{u}) \;-\; \boldsymbol{\nabla} KE \;+\; \mathbf{b} \;+\; \frac{1}{\rho_0}\boldsymbol{\nabla} \cdot \boldsymbol{\tau} \;-\; \frac{1}{\rho_0}\boldsymbol{\nabla} p, \tag{4}$$

This formulation is advantageous in that it takes the same form in all coordinate systems i.e., there is no advective derivative of the coordinate system, which shows up as a "metric" tendency in the flux-form momentum equations (see MITgcm manual section 2.15). Dynamical insights into forcing variations driving circulation changes are most easily achieved by retaining separation between different momentum sources, sinks and redistributions (i.e., distinguishing between wind and dissipative stresses, separating pressure gradient forces from advection etc.) Separation of terms also helps to understand the momentum terms coded in ASTE_R1 and determine the diagnostic output required for momentum budget closure.

3 MITgcm Discretized Horizontal Momentum Equations

Individual contributions to the Eulerian acceleration of the horizontal flow in Eq. (4) may now be written (slightly different notation to that in manual section 2.14) as follows, in a way that identifies horizontal terms in Eq. (4) with numerical terms in the MITgcm as follows:

$$\begin{aligned} G_u = G_u^{fcori} &+ G_u^{Adv1} + G_u^{Adv2} + G_u^{Adv3} \ ... \\ &+ G_u^{horiz-diss} + G_u^{vert-diss} + G_u^{Ext} \ ... \\ &+ G_u^{dPHdx} + G_u^{dP_{sfc}dx}, \end{aligned} \tag{5}$$

$$\begin{aligned} G_v = G_v^{fcori} &+ G_v^{Adv1} + G_v^{Adv2} + G_v^{Adv3} \ ... \\ &+ G_v^{horiz-diss} + G_v^{vert-diss} + G_v^{Ext} \ ... \\ &+ G_v^{dPHdy} + G_v^{dP_{sfc}dy}, \end{aligned} \tag{6}$$

The first RHS term describes momentum redistribution by the Coriolis force:

$$\begin{bmatrix} G_u^{fcori} \\ G_v^{fcori} \end{bmatrix} = \begin{bmatrix} -(f\hat{\mathbf{k}} \times \mathbf{u}) \cdot i \\ -(f\hat{\mathbf{k}} \times \mathbf{u}) \cdot j \end{bmatrix} = \begin{bmatrix} fv \\ -fu \end{bmatrix} = \begin{bmatrix} (1/\Delta x_c)(\overline{f_\zeta}^j / \overline{\overline{h_s}^j}^i) \overline{\overline{\Delta x_g h_s v^j}}^i \\ -(1/\Delta y_c)(\overline{f_\zeta}^i / \overline{\overline{h_w}^i}^j) \overline{\overline{\Delta y_g h_w u^i}}^j \end{bmatrix}, \tag{7}$$

where the traditional approximation has been made to retain only the locally vertical component of the rotation vector. The discretization (on an Arakawa C-grid, Fig. 1) eliminates boundary points from the computation of the Coriolis force in coastal cells (`useJamartWetPoints = TRUE`) and conserves potential enstrophy (see */pkg/mom_vecinv/mom_vi_coriolis.F*). The overbar indicates averaging (along the indicated direction). Here, the Coriolis parameter, f_ζ, is defined at the vorticity points (Fig. 1). For example, to compute the u-momentum tendency from Coriolis, f_ζ (on the cell corners) is averaged in y, onto the u-velocity point. This is then multiplied with the v-velocity, which has first been moved onto the tracer point (cell center), then onto the u-velocity point with weighted averages. For the remaining terms in Eqs. (5) & (6) the discretization implemented in ASTE_R1 is given below. The reader is referred to the manual (or the relevant S/R given in Table 1) for other options and additional explanation. We'll focus on clarifying term groupings in the diagnostics and closing the budget.

The second and third terms on the RHS term of Eqs. (5) & (6) are the parts of the tendency from inertia (Eq. (3)) not associated with the KE gradient:

$$-\boldsymbol{\zeta} \times \mathbf{u} = \begin{bmatrix} -\zeta_2 w + \zeta_3 v \\ \zeta_1 w - \zeta_3 u \\ -\zeta_1 v + \zeta_2 u \end{bmatrix}, \tag{8}$$

where subscript $1, 2, 3$ denote i, j, k vector components respectively (following MITgcm manual notation). Depending on the choice of advection scheme, the part involving the horizontal flux of the vertical component of relative vorticity can be combined with the Coriolis tendency (Eq. (7)). In the manual, this component is referred to as the "non-linear Coriolis term":

$$\begin{bmatrix} G_u^{Adv1} \\ G_v^{Adv1} \end{bmatrix} = \begin{bmatrix} \zeta_3 v \\ -\zeta_3 u \end{bmatrix} = \begin{bmatrix} (1/\Delta x_c)(\overline{\zeta_3/h_\zeta}^j) \overline{\overline{\Delta x_g h_s v^j}}^i \\ -(1/\Delta y_c)(\overline{\zeta_3/h_\zeta}^i) \overline{\overline{\Delta y_g h_w u^i}}^j \end{bmatrix}, \tag{9}$$

The remaining part – due to vertical shear in the horizontal velocity – is referred to as the "shear term":

$$\begin{bmatrix} G_u^{Adv2} \\ G_v^{Adv2} \end{bmatrix} = \begin{bmatrix} -\zeta_2 w \\ \zeta_1 w \end{bmatrix} = \begin{bmatrix} -1/(A_w \Delta r_f h_w) \overline{\overline{A_c w}^i \delta_k u}^k \\ -1/(A_s \Delta r_f h_s) \overline{\overline{A_c w}^j \delta_k v}^k \end{bmatrix}, \tag{10}$$

where the non-hydrostatic contribution has been neglected and `upwindShear = FALSE`. Note that the negative sign on the discretised G_v^{Adv2} is from $+\partial_k v = -\zeta_1$.
The fourth RHS term in Eqs. (5) & (6) is the advective tendency due to horizontal gradients in kinetic energy:

$$\begin{bmatrix} G_u^{Adv3} \\ G_v^{Adv3} \end{bmatrix} = \begin{bmatrix} -\partial(KE)/\partial x \\ -\partial(KE)/\partial y \end{bmatrix} = \begin{bmatrix} -(1/\Delta x_c)\delta_i KE \\ -(1/\Delta x_y)\delta_j KE \end{bmatrix}, \tag{11}$$

where δ is the difference in the indicated direction (e.g., $\delta_i KE = KE(i,j,k) - KE(i-1,j,k)$. In the code this term is referred to as the gradient in Bernoulli potential ($B = KE + \phi$), but pressure gradients are handled separately in practice.

The fifth RHS term in Eqs. (5) & (6) is the momentum tendency from horizontal dissipation, combining contributions from the Laplacian and biharmonic viscosity:

$$\begin{bmatrix} G_u^{horiz-diss} \\ G_v^{horiz-diss} \end{bmatrix} = \begin{bmatrix} \frac{1}{\Delta x_c}\delta_i(\mu_D D - \mu_{D4} D^*) \; - \; \frac{1}{\Delta y_u h_w}\delta_j h_\zeta(\mu_\zeta \zeta - \mu_{\zeta 4}\zeta^*) \\ \frac{1}{\Delta x_v h_s}\delta_i h_\zeta(\mu_\zeta \zeta - \mu_\zeta \zeta^*) \; + \; \frac{1}{\Delta y_c}\delta_j(\mu_D D - \mu_{D4} D^*) \end{bmatrix}. \tag{12}$$

Here the horizontal velocity divergence is given by:

$$D = \boldsymbol{\nabla} \cdot \mathbf{u} = \frac{1}{A_c h_c}(\delta_i \Delta y_g h_w u + \delta_j \delta x_g h_s v). \tag{13}$$

The divergence of the Laplacian is given by:

$$D^* = \boldsymbol{\nabla} \cdot \boldsymbol{\nabla}^2 \mathbf{u} = \frac{1}{A_c h_c}(\delta_i \Delta y_g h_w \boldsymbol{\nabla}^2 u + \delta_j \delta x_g h_s \boldsymbol{\nabla}^2 v), \tag{14}$$

and:

$$\zeta^* = \frac{1}{A_\zeta}(\delta_i \Delta y_c \boldsymbol{\nabla}^2 v - \delta_j \delta x_c \boldsymbol{\nabla}^2 u), \tag{15}$$

where this discretization is chosen to conserve divergence and PV (thickness-weighted relative vorticity) and dissipate energy, enstrophy and divergence squared. See the call in *mom_vi_hdissip.F* to *mom_vi_del2uv.F* and following calls to *mom_calc_hdiv.F* (output `dStar`) and *mom_calc_relvort3.F* (output `zstar`). Contributions from side drag and bottom drag (the latter including both application of the no slip boundary condition and additional friction) are added after these terms have been computed. In ASTE the viscous coefficients, $(\mu_D, \mu_\zeta, \mu_{D4}, \mu_{\zeta 4})$, are background values augmented by the addition of a time- and space-dependent eddy viscosity, determined locally from gradients in vorticity and divergence, following Leith [1996], Fox-Kemper and Menemenlis [2008].

The sixth term in Eqs. (5) & (6) is the momentum tendency from vertical dissipation:

$$\begin{bmatrix} G_u^{vert-diss} \\ G_v^{vert-diss} \end{bmatrix} = \begin{bmatrix} \frac{1}{\Delta r_f h_w}\delta_k \left(A_v \frac{1}{\Delta r_c}\delta_k u \right) \\ \frac{1}{\Delta r_f h_s}\delta_k \left(A_v \frac{1}{\Delta r_c}\delta_k v \right) \end{bmatrix}, \tag{16}$$

where - in ASTE_R1 - the vertical diffusivity, A_v, is modified by the KPP scheme (Large et al. [1994]). The seventh term in Eqs. (5) & (6) is the momentum tendency from wind forcing:

$$\begin{bmatrix} G_u^{Ext} \\ G_v^{Ext} \end{bmatrix} = \begin{bmatrix} \frac{\tau^x}{\Delta r_f h_w} \\ \frac{\tau^y}{\Delta r_f h_s} \end{bmatrix}, \tag{17}$$

defined only in the surface layer where the wind stress $[\tau^x, \tau^y]$ is felt, although other terms can contribute if using a configuration that is different to ASTE_R1. The final two RHS terms in Eqs. (5) & (6) are momentum tendencies due to horizontal pressure gradients:

$$\begin{bmatrix} G_u^{dPHdx} \\ G_v^{dPHdy} \end{bmatrix} = \begin{bmatrix} -(1/\rho_0 \Delta x_c)\delta_i(P_{hyd} + P_{atm}) \\ -(1/\rho_0 \Delta x_y)\delta_j(P_{hyd} + P_{atm}) \end{bmatrix}. \tag{18}$$

where P_{hyd} is the hydrostatic pressure and P_{atm} is the atmospheric loading. Pressure gradients due to the free surface displacement are isolated in the final term:

$$\begin{bmatrix} G_u^{dP_{sfc}dx} \\ G_v^{dP_{sfc}dy} \end{bmatrix} = \begin{bmatrix} -(g/\Delta x_c)\delta_i \eta \\ -(g/\Delta x_y)\delta_j \eta \end{bmatrix}, \tag{19}$$

where η is the free surface displacement (m) and g is gravity.

4 MITgcm Required Diagnostic Output

Each model execution automatically writes a list (file *available_diagnostics.log*) of available diagnostics to the output directory. For each pre-defined diagnostic the name, array dimension, grid location and physical unit is given, along with a simple description (see also MITgcm manual, Section 9.1). **For ASTE_R1, the momentum budget can be closed using available diagnostics, with the addition of diagnostics for the implicit vertical viscosity and surface pressure gradient**. There are also other diagnostic metrics available that are insightful, but not required to close the ASTE_R1 momentum budget. All relevant diagnostics are given in Table 1, split into 4 groups:

- Group 1: Required pre-defined diagnostics,
- Group 2: Required user-defined diagnostics,
- Group 3: Relevant (not required) pre-defined diagnostics.
- Group 4: Relevant (not required) user-defined diagnostics.

Additional explanation is given only where necessary below. Note that this description is specific to the choices of available numerical algorithms made in ASTE_R1. For example, pre-defined strain and tension diagnostics may also be useful for understanding the dissipation when it is not formulated in terms of vorticity and divergence (as in ASTE_R1).

A closed momentum budget at [u,v] points on the k-th model level of ASTE_R1 is given by:

$$\begin{aligned}\frac{\texttt{TOTUTEND(i,j,k)}}{86400} = \texttt{Um\_Cori(i,j,k)} &+ \Big(\texttt{Um\_Advec(i,j,k)} - \texttt{Um\_Cori(i,j,k)}\Big) \\ &+ \texttt{Um\_Diss(i,j,k)} + \texttt{Um\_Ext(i,j,k)} + \texttt{Um\_dPHdx(i,j,k)} \\ &+ \texttt{Um\_dPsdx(i,j,k)} + \texttt{AB\_gU(i,j,k)} + \texttt{Um\_Impl(i,j,k)}\end{aligned} \tag{20}$$

$$\begin{aligned}\frac{\texttt{TOTVTEND(i,j,k)}}{86400} = \texttt{Vm\_Cori(i,j,k)} &+ \Big(\texttt{Vm\_Advec(i,j,k)} - \texttt{Vm\_Cori(i,j,k)}\Big) \\ &+ \texttt{Vm\_Diss(i,j,k)} + \texttt{Vm\_Ext(i,j,k)} + \texttt{Vm\_dPHdy(i,j,k)} \\ &+ \texttt{Vm\_dPsdy(i,j,k)} + \texttt{AB\_gV(i,j,k)} + \texttt{Vm\_Impl(i,j,k)}\end{aligned} \tag{21}$$

Note that all diagnostics are output on the correct grid (i.e., the [u,v]-momentum tendencies are output at the [u,v]-velocity points, respectively). For ASTE_R1 it is useful to decompose tendencies from advection and dissipation; we define additional diagnostics to allow this. For consistency, new diagnostics added to the model output are also written as RHS tendencies (i.e., such that they can all be added to obtain the Eulerian velocity tendency). With this extra output, the u-momentum decomposition is given as follows:

$$\texttt{Um\_Advec} = \texttt{Um\_Cori} + \texttt{Um\_AdvZ3} + \texttt{Um\_AdvRe} + \texttt{Um\_dKEdx} \tag{22}$$

$$\texttt{Um\_Diss} = \texttt{Um\_Diss2} + \texttt{Um\_Diss4} + \texttt{UBotDrag} + \texttt{USidDrag} \tag{23}$$

and an analogous decomposition can be written for v-momentum tendencies. Additional sources may contribute for model configurations with different CPP options (e.g., additional contributions may be made to [Um_Ext,Vm_Ext] which is why we output [Um_Wind,Vm_Wind] separately, although this is currently the only contribution in ASTE_R1).

GROUP	NAME	DESCRIPTION
1	[TOTUTEND,TOTVTEND]	[u,v] total Eulerian tendencies
1	[Um_dPHdx,Vm_dPHdy]	[u,v] tend. from *hydrostatic* pressure gradient
1	[Um_Cori,Vm_Cori]	[u,v] tend. from Coriolis (in [Um_Advec,Vm_Advec])
1	[Um_Advec,Vm_Advec]	[u,v] tend. from inertia and Coriolis
1	[Um_Ext,Vm_Ext]	[u,v] tend. from external forcing (just wind in ASTE_R1)
1	[AB_gU,AB_gV]	[u,v] tend. increment from Adams-Bashforth timestepping
1	[VISrI_Um,VISrI_Vm]	[u,v] vol-integrated tend. from vertical viscous flux
2	[Um_dPsdx,Vm_dPsdy]	[u,v] tend. from free surface displacement
2	[Um_Impl,Vm_Impl]	[u,v] tend. from vertical viscous flux
3	[UBotDrag,VBotDrag]	[u,v] tend. from bottom drag (in [Um_Diss,Vm_Diss])
3	[USidDrag,VSidDrag]	[u,v] tend. from side drag (in [Um_Diss,Vm_Diss])
3	[Um_AdvZ3,Vm_AdvZ3]	[u,v] tend. from vorticity advection (in [Um_Advec,Vm_Advec])
3	[Um_AdvRe,Vm_AdvRe]	[u,v] tend. from vertical shears (in [Um_Advec,Vm_Advec])
3	momVort3	vertical component of vorticity (no slip BC applied)
3	momKE	kinetic energy
3	momHDiv	horizontal divergence
4	[Um_Wind,Vm_Wind]	[u,v] tend. from surface wind forcing (in [Um_Ext,Vm_Ext])
4	[Um_CorNL,Vm_CorNL]	[u,v] tend. from non-traditional Coriolis
4	[Um_dKEdx,Vm_dKEdy]	[u,v] tend. from KE gradients (in [Um_Advec,Vm_Advec])
4	[Um_Diss2,Vm_Diss2]	[u,v] tend. from harmonic viscosity (in [Um_Diss,Vm_Diss])
4	[Um_Diss4,Vm_Diss4]	[u,v] tend. from biharmonic viscosity (in [Um_Diss,Vm_Diss])

Table 1: MITgcm diagnostic output relevant for momentum budget evaluation. The 4 groups delineate (1) pre-defined diagnostics required for momentum budget closure, (2) user-defined diagnostics required for momentum budget closure, (3) pre-defined diagnostics that are relevant but not required for momentum budget closure, and (4) user-defined diagnostics that are relevant but not required for momentum budget closure. Here our definition of "required" diagnostics is specific to ASTE_R1. In green, we highlight an important difference for ASTE_R1 configured with a nonlinear versus linear free surface (since code for re-running both configurations has been made available): [Um_Impl,Vm_Impl] are required for the former, but [VISrI_Um,VISrI_Vm] will suffice for the latter. "Pre-defined" diagnostics are those that are already coded in checkpoint 65q (or later) of the MITgcm (i.e., listed in available_diagnostics.log). Though they are not included in ASTE_R1, our list of "User-defined" diagnostics may be insightful and require very minor code changes, as described in sections 5 & 6.

NAME	UNITS	LOCATION	KEY S/R
[TOTUTEND,TOTVTEND]	**ms^{-1}day^{-1}**	[u,v]	*mom_vecinv.F* *timestep.F*
[Um_dPHdx,Vm_dPHdy]	ms^{-2}	[u,v]	*calc_phi_hyd.F* *calc_grad_phi_hyd.F*
[Um_Cori,Vm_Cori]	ms^{-2}	[u,v]	*mom_vi_coriolis.F*
[Um_Advec,Vm_Advec]	ms^{-2}	[u,v]	*mom_vi_coriolis.F* *mom_vi_*_vertshear.F* *mom_vi_*_grad_KE.F* *mom_vi_*_coriolis.F*
[Um_Diss,Vm_Diss]	ms^{-2}	[u,v]	*mom_calc_visc.F* *mom_calc_del2uv.F* *mom_calc_hdiv.F* *mom_calc_relvort3.F* *mom_vi_hdissip.F*
[Um_Ext,Vm_Ext]	ms^{-2}	[u,v]	*apply_forcing_*.F* *external_forcing_surf.F* *external_fields_load.F*
[AB_gU,AB_gV] [VISrI_Um,VISrI_Vm]	ms^{-2} **m^{4}s^{-2}**	[u,v] **w-level** [u,v]	*adams_bashforth2.F* *impldiff.F* *calc_viscosity.F* *kpp_calc_viscosity.F*
[Um_dPsdx,Vm_dPsdx]	ms^{-2}	[u,v]	*momentum_correction_step.F* *calc_grad_phi_surf.F*
[Um_Impl,Vm_Impl]	ms^{-2}	[u,v]	*impldiff.F* *calc_viscosity.F* *kpp_calc_viscosity.F*
[UBotDrag,VBotDrag]	ms^{-2}	[u,v]	*mom_*_botdrag.F*
[USidDrag,VSidDrag]	ms^{-2}	[u,v]	*mom_*_sidedrag.F*
[Um_AdvZ3,Vm_AdvZ3]	ms^{-2}	[u,v]	*mom_vi_*_coriolis.F*
[Um_AdvRe,Vm_AdvRe]	ms^{-2}	[u,v]	*mom_vi_*_vertshear.F*
momVort3	s^{-1}	ζ	*mom_calc_relvort3.F*
momKE	m^{2}s^{-2}	p	*mom_calc_ke.F*
momHDiv	s^{-1}	p	*mom_calc_hdiv.F*
[Um_Wind,Vm_Wind]	ms^{-2}	[u,v]	*apply_forcing_*.F* *external_forcing_surf.F* *external_fields_load.F*
[Um_CorNL,Vm_CorNL]	ms^{-2}	[u,v]	*mom_*_coriolis_nh.F*
[Um_dKEdx,Vm_dKEdy]	ms^{-2}	[u,v]	*mom_vi_*_gradKE.F*
[Um_Diss2,Vm_Diss2]	ms^{-2}	[u,v]	*mom_vi_hdissip.F*
[Um_Diss4,Vm_Diss4]	ms^{-2}	[u,v]	*mom_vi_hdissip.F*

Table 2: Units, location and key subroutines for each MITgcm diagnostic related to inspection of the closed momentum budget in ASTE_R1, as given in Table 1. Subroutines given in red are those edited (from MITgcm c65q) to enable the output of the associated diagnostic, with the exception of `mom_vecinv.F`, which is modified to output [Um_dKEdx,Vm_dKEdy] after these terms have been computed in `mom_vi_*_gradKE.f`. In addition to these routines, `mom_diagnostics_init.F` must also be edited (see Section 5).

4.1 Required pre-defined Diagnostics

[TOTUTEND,TOTVTEND] is the total Eulerian acceleration. It should be checked that this (divided by 1 day in seconds) is equal to the sum of all tendency terms to within numerical round off error (see Eqs. (20), (21)).

[Um_dPHdx,Vm_dPHdy] is the tendency due to horizontal gradients in the hydrostatic pressure. The pressure is first computed by integrating hydrostatic balance and setting $\phi(z = 0) = 0$ in S/R *calc_phi_hyd.F*. The atmospheric surface pressure (phi0surf) is then added before the gradient is computed in *calc_grad_phi_hyd.F*. In ASTE_R1 phi0surf is uniformly zero. Importantly, for either a linear or nonlinear formulation of the free surface, the pressure gradient contribution from the sea surface height displacement is *not* included in the computation of [Um_dPHdx,Vm_dPHdy]. For this reason, we defined an extra diagnostic [Um_dPsdx,Vm_dPsdy] for this contribution, which must also be output to close the ASTE_R1 momentum budget (see next sub-section).

[Um_Cori,Vm_Cori] is the momentum redistribution by the Coriolis force (planetary vorticity alone). If one chooses to redistribute momentum by the absolute vorticity (flag useAbsVorticity = .TRUE.,), [Um_Cori,Vm_Cori] will also contain momentum redistribution by the relative vorticity i.e., tendencies in Eqs. (7) and (9) are combined. In ASTE_R1 useAbsVorticity = .FALSE.,. Unless one chooses the CD-scheme [Um_Cori,Vm_Cori] are included in [Um_Advec,Vm_Advec]. In ASTE_R1 the Coriolis acceleration is included in the advective tendency.

[Um_Advec,Vm_Advec] is the momentum redistribution by inertia. In ASTE_R1, there are 4 distinct terms contributing to this tendency: (a) momentum redistribution by relative vorticity (Eq. (9)), (b) momentum redistribution by vertical shear terms (Eq. (10)), (c) kinetic energy gradients (Eq. (11)), not including the pressure anomaly), and (d) momentum redistribution by planetary vorticity ([Um_Cori,Vm_Cori]).

[Um_Diss,Vm_Diss] is the momentum tendency from dissipation arising from all explicit dissipation terms. In ASTE_R1 there are 4 distinct contributions (a) laplacian viscosity, (b) biharmonic viscosity, (c) side drag and (d) bottom drag. The drag is applied only in the cells neighboring topography. An additional contribution can arise from no slip conditions imposed on flow under ice shelves, but this is not included in ASTE_R1. ASTE_R1 includes an additional dissipation via implicit viscosity (implicitViscosity=.TRUE., in *input/data*). This forcing is not included in [Um_Diss,Vm_Diss] but can be obtained from the predefined diagnostic [VISrI_Um,VISrI_Vm] if a linear free surface is used. If a nonlinear free surface with vertically rescaled (z*) coordinates is employed, our diagnostic [Um_Impl,Vm_Impl] should be written out during the run (see notes on [Um_Impl,Vm_Impl] below).

[Um_Ext,Vm_Ext] is the horizontal momentum tendency from external forcing. In ASTE_R1 this is from wind forcing alone, but in other configurations, there may be additional contributions, for example from relaxation boundary conditions or eddy stresses enabled via the GM parameterization.

[AB_gU,AB_gV] is the acceleration adjustment to account for use of the Adams-Bashforth time-stepping scheme. This scheme computes tendencies at time levels n and $n-1$ separately and then extrapolates to $n+1/2$:

$$G^{n+1/2} = (3/2 + \epsilon_{AB})G^n \; - \; (1/2 + \epsilon_{AB})G^{n-1} \tag{24}$$

All prognostic variables (c, including velocity components and all tracers) are then stepped forward using the extrapolated tendency ($c^{n+1} = f(G^{n+1/2})$). ϵ_{AB} is small but necessary to achieve numerical stability. From *adams_bashforth2.F* the associated tendencies are:

$$[\texttt{AB\_gU},\texttt{AB\_gV}] = \left[\frac{\epsilon_{AB}}{2} * (G_u^n - G_u^{n-1}), \frac{\epsilon_{AB}}{2} * (G_v^n - G_v^{n-1})\right]. \tag{25}$$

[VISrI_Um,VISrI_Vm] is the volume-integrated momentum tendency from implicit vertical advection and viscosity (also referred to as momentum diffusion). The advective contribution is computed in `mom_[u,v]_implicit_r.F` and is nonzero only by setting `momImplVertAdv = .TRUE.`, in `data` (not the default nor the case in ASTE_R1). The diffusive (viscous) component is computed in `impldiff.F` and is nonzero only by setting `ImplViscosity=.TRUE.` in `data` (as in ASTE_R1 but not by default). Note that `VISrI_[U,V]m` are horizontally aligned with $[u, v]$, respectively, but are defined at the w-level and therefore need to be averaged on to the mid-level. Also, we need to divide by the cell volume ([`rAw*drF(k)*hFacW, rAs*drF(k)*hFacS`], centered on $[u, v]$, respectively) to obtain units of ms^{-2}.

Additional notes important for computation: `VISrI_Um(k=1)` should be zero; `VISrI_Um(k=Nr+1)` is not stored in the output file but it is also zero. For ASTE_R1 integrated with a nonlinear free surface and z* coordinates, the user should output `[Um_Impl,Vm_Impl]` instead of using this diagnostic, to account for time-varying cell volume (from z*). We have included it here, because the files required for re-running ASTE_R1 have been released with the option to switch to linear free surface [Nguyen et al. 2021a], for which this diagnostic will suffice. We have not made a diagnostic for the momentum tendency from implicit vertical advection (which can contribute to `VISrI_[U,V]m`) and so the user should do this if they compile with nonlinear free surface, z* coordinates and `momImplVertAdv = .TRUE.`. Finally, discussion on the MITgcm support list suggests `impldiff.F` is anomalously expensive (due to poor cache efficiency) on some platforms and may be retired in the future. The reader using checkpoints later than c65q should check whether the momentum tendency from implicit vertical viscosity (diffusion) has been rolled into the implicit vertical advection subroutine (`mom_[u,v]_implicit_r.F`), which sounds like the plan at the time of writing.

4.2 Required user-defined diagnostics

[Um_dPsdx,Vm_dPsdy] is the pressure gradient created by the displacement of the free surface. It is *not* included in the hydrostatic pressure gradient force diagnostic ([Um_dPHdx,Vm_dPHdy]). Importantly, [Um_dPsdx,Vm_dPsdy] is a 2D field, but this horizontal gradient projects onto the horizontal pressure gradient force in all underlying layers (i.e., it contributes to the momentum tendency at all model levels). We output this tendency in *momentum_correction_step.F*.

[Um_Impl,Vm_Impl] is the momentum tendency from the implicit vertical viscosity (momentum diffusivity). Note that offline computation of these terms from pre-defined diagnostics for the volume-integrated tendency from implicit vertical viscosity [VISrI_Um,VISrI_Vm] is sufficient when using a linear free surface [not the case for ASTE_R1 but can be selected for re-runs, see Nguyen et al. 2021a]. These pre-defined terms are given on the $[u, v]$ latitudes and longitudes, but at the w-level and therefore need to be averaged on to the mid-level and divided by the cell volume ([`rAw*drF(k)*hFacW, rAs*drF(k)*hFacS`], respectively) to obtain units of ms^{-2}:

$$\texttt{Um\_Impl(i,j,k)} = \left(\frac{\texttt{VISrI\_Um(i,j,k+1) - VISrI\_Um(i,j,k)}}{\texttt{[rAw(i,j)*drF(k)*hFacW(i,j,k)]}} \right) \tag{26}$$

$$\texttt{Vm\_Impl(i,j,k)} = \left(\frac{\texttt{VISrI\_Vm(i,j,k+1) - VISrI\_Vm(i,j,k)}}{\texttt{[rAs(i,j)*drF(k)*hFacS(i,j,k)]}} \right) \tag{27}$$

Both `VISrI_Um(k=1)` and `VISrI_Um(k=Nr+1)` should be set to zero; note that the former is hard-coded in `impldiff.F` and the latter is not included in the output. For all configurations using non-linear free surface with z* coordinates (including ASTE_R1), it is important to note that the open cell fraction `hfac*` is also time-evolving, since the z* vertical coordinate ($z* = [z - \eta/H + \eta] * H$) stretches to follow the physical free surface, avoiding vanishing layers [Adcroft and Campin 2004]. Although the stretching is small (because $|\eta| << |H|$), division by cell volume cannot be performed offline for accurate closure of the momentum budget, so we have introduced the diagnostics `[Um_Impl,Vm_Impl]` to perform the computation online.

4.3 Relevant (not required) pre-defined diagnostics

[USidDrag, VSidDrag] is the contribution of the lateral boundary condition (no-slip in ASTE_R1 with shear applied to the coastal cells only) to the total dissipative momentum tendency [`Um_Diss`, `Vm_Diss`].

[UBotDrag,VBotDrag] is the contribution of the bottom (no-slip) boundary condition (shear applied in the deepest cell only) and an additional frictional force to the total dissipative momentum tendency [`Um_Diss`,`Vm_Diss`].

[Um_AdvZ3,Vm_AdvZ3] is the contribution of momentum redistribution by relative vorticity to the total inertial momentum tendency [`Um_Advec`, `Vm_Advec`]. Note it looks a bit confusing in the source (S/R *mom_vecinv.F*) because it is computed by calling a Coriolis S/R (*mom_vi_*_coriolis.F*). Note however that this term (Eq. (9)) has the same form as the traditional Coriolis term (Eq. (7)) and the input arguments are different for computing this diagnostic (so that only ζ_3 operates on $\mathbf{u}$).

[Um_AdvRe,Vm_AdvRe] is the contribution of the vertical shear terms (Eq. (10)) to the total inertial momentum tendency [`Um_Advec`, `Vm_Advec`].

momVort3 and momKE are the vertical component of the relative vorticity, ζ_3, and the kinetic energy, KE, respectively. These fields are useful for understanding [`Um_Advec`,`Vm_Advec`] and they have also proved useful in the past for understanding the rotational momentum force functions.

momHDiv is the divergence of horizontal velocity, computed so that it can be damped at the grid scale (along with gradients in relative vorticity) in [`Um_Diss`,`Vm_Diss`] using modified Leith viscosity.

4.4 Relevant (not required) user-defined diagnostics

[Um_Wind,Vm_Wind] is the tendency due to the wind alone. This is currently included in [`Um_Ext`,`Vm_Ext`], but as above, it may be advantageous to output it separately because - with different CPP options - other terms may contribute here.

[Um_CorNL,Vm_CorNL] is the tendency due to the non-traditional ($2\Omega \cos\phi$) component of the Coriolis force. This is currently included in [Um_Advec,Vm_Advec]. This tendency is computed in S/R [*mom_u_coriolis_nh.F*,*mom_v_coriolis_nh.F*] and output directly to be added to the incrementally updated acceleration in [S/R *mom_vecinv.F*], so we have added a call here to fill the diagnostic. This could be particularly interesting to examine the forces driving equatorial crossing of the overturning circulation if we run quasi-hydrostatic or non-hydrostatic code [e.g., Stewart and Dellar 2012]. If assuming hydrostatic balance, as for ASTE_R1, both the cosine term contributing to the vertical acceleration and the cosine term contributing to the zonal acceleration are dropped and this diagnostic will be all zero. Note even if you have the flag `use3dCoriolis = .TRUE.` in `input/data` (as accidentally left in ASTE_R1) it will be switched to `.FALSE.` in ini_parms.F if you do not also add explicit switch to quasi-hydrostatic (`quasiHydrostatic = .TRUE.`) or non-hydrostatic (`nonHydrostatic = .TRUE.`) code in `input/data`.

[Um_dKEdx,Vm_dKEdy] is the tendency from the kinetic energy gradient. This is currently included in [Um_Advec,Vm_Advec]. The 3 other distinct contributions are written separately and we have also output this term. This tendency is computed in S/R [*mom_vi_u_grad_ke.F*,*mom_vi_v_grad_ke.F*] and output directly to be added to the incrementally updated acceleration in S/R *mom_vecinv.F*, so we have added a call here to fill the diagnostic.

[Um_Diss2,Vm_Diss2] and [Um_Diss4,Vm_Diss4] are the tendencies due to the laplacian and biharmonic dissipation, respectively. These are currently included in [Um_Diss,Vm_Diss] but it is interesting to look at their contribution separately (partly because - as will be shown in Part II - the rotational momentum tendencies are so strongly shaped by the choice of accompanying boundary condition). For ASTE_R1 [Um_Diss,Vm_Diss] are computed in S/R mom_vecinv.F and the harmonic and biharmonic contributions are calculated by calling S/R *mom_vi_hdissip.F*. We have added the call to fill the diagnostics here. If not using ASTE_R1, check this because the components of the dissipative tendency will not add up to the total ([Um_Diss,Vm_Diss]) for some CPP options (e.g., if (useSmag3D=.TRUE.,) and/or if (useShelfIce=.TRUE.,))

5 How to Use the Diagnostics Package

The reader is referred to the official MITgcm manual, Section 9.1, for clear guidance on using MITgcm diagnostic output. For simply re-running ASTE_R1 to output the diagnostic terms described in Table 2, you can obtain the necessary code and namelist modifications from our group github or the Corral storage resource at the Texas Advanced Computing Center (see section 7). For those wishing to create additional new diagnostics, the necessary steps can be briefly summarized as follows:

1. Enable the diagnostics package in *code/packages.conf* prior to compiling.
2. Set `useDiagnostics = .TRUE.`, in *input/data.pkg*.
3. Add desired diagnostics to *inputs/data.diagnostics*. In this file one can also specify (i) the output frequency (seconds), (ii) whether snapshots (frequency < 0) or time-averages (frequency > 0) are required, (iii) the phase offset (seconds) of the output (i.e., whether it the time-average is output in the middle of the averaging period), (iv) the filename for the output diagnostic.

To add a new diagnostic, for example to write the momentum tendency from laplacian viscosity ([`Um_Diss2,Vm_Diss2`]) to the diagnostic output in the current model run, the following changes should be made prior to compilation:

1. Copy *mom_common/mom_diagnostics_init.F* to *code/*. Add meta and call to add [`Um_Diss2,Vm_Diss2`], mirroring structure for existing momentum diagnostics:

```
diagName = 'Um_DISS2  '
diagTitle = 'U momentum tendency from harmonic visc alone'
diagCode = 'UUR     MR'
diagMate = diagNum + 2
CALL DIAGNOSTICS_ADDTOLIST( diagNum,
I diagName, diagCode, diagUnits, diagTitle, diagMate, myThid )
diagName = 'Vm_Diss2  '
diagTitle = 'V momentum tendency from harmonic visc alone'
diagCode = 'VVR     MR'
diagMate = diagNum
CALL DIAGNOSTICS_ADDTOLIST( diagNum,
I diagName, diagCode, diagUnits, diagTitle, diagMate, myThid )
```

   Note that `diagName` should be defined with 8 characters (i.e., be careful to include blank space or the run will crash at the start with `ABNORMAL END: S/R DIAGNOSTICS_SET_POINTERS`).

2. Identify where the diagnostic is (or needs to be) computed and copy this S/R to code/. The annotated call tree in section 6 will hopefully be helpful here. Then edit this S/R as necessary, including adding a call to *DIAGNOSTICS_FILL* or *DIAGNOSTICS_SCALE_FILL*, mirroring structure for existing diagnostics.

3. Check in *code/DIAGNOSTICS_SIZE.h* that the diagnostic common block will be big enough to accommodate additional user defined 2D/3D diagnostics. Increase `numdiags` if necessary.

6 Diagnostic Call Tree/Flow Chart

Here we outline the call tree important for momentum closure, focusing on where terms are computed, accumulated,and written to file. Details of the calculations are skipped here, avoiding repetition of information given in the sections above and keeping clearer visualization of the flow between important S/R.

```
forward_step
|
|--dynamics
|      |
|      |INITIALISE LOCAL VARS FOR DISSIPATIVE TENDENCY, PGF AND FULL TENDENCY
|      |DO k = 1,Nr
|      | DO j = 1-Oly,sNy+OLy
|      |  DO i = 1-Olx,ONx+OLx
|      |   gU(i,j,k,bi,bj) = 0.DO
|      |   gV(i,j,k,bi,bj) = 0.DO
|      |  ENDDO
|      | ENDDO
|      |ENDDO
|      |DO j = 1-Oly,sNy+OLy
|      | DO i = 1-Olx,ONx+OLx
|      |  phiSurfX(i,j) = 0.DO
|      |  phiSurfY(i,j) = 0.DO
|      |  guDissip(i,j) = 0.DO
|      |  gvDissip(i,j) = 0.DO
|      | ENDDO
|      |ENDDO
|      |
|      |START BY COMPUTING HYDROSTATIC AND SFC PGF. THESE ARE APPLIED TO ACCELERATE THE FLOW MUCH LATER IN S/R timestep.F
|      |--- calc_grad_phi_surf explicit part of ∇_h p_sfc, output = [phiSurfX,phiSurfY]
|      |                                To account for full surface PGF we output tendency diagnostic in S/R momentum_correction_step, not here.
|      |
|      |--- calc_viscosity compute net VERTICAL viscosity, output = [kappaRU,kappaRV], used way later at the end of S/R timestep.F
|      |      |--kpp_calc_visc include contribution from KPP
|      |      o
|      |
|      |START OF DYNAMICS LOOP OVER VERTICAL LEVELS
|      |do k = 1,Nr
|      |--- calc_phi_hyd integrate hydrostatic balance for level k
|      |      |--calc_grad_phi_hyd compute ∇_h p_hyd for level k, output=[dPhyHydX,dPhiHydY]
|      |      o
|      |tmpFac = -1.  d 0 change sign to save as RHS tendency
|      |CALL DIAGNOSTIC_SCALE_FILL( dPhiHydX, tmpFac, 1,
|      |                   'Um_dPHdx',k,1,2,bi,bj,myThid )
|      |CALL DIAGNOSTIC_SCALE_FILL( dPhiHydY, tmpFac, 1,
|      |                    'Vm_dPHdy',k,1,2,bi,bj,myThid )
|      |
```

```
|       |CALL WRAPPER FOR MOMENTUM TENDENCY CALCULATIONS
|       |--- mom_vecinv if not using fluxform, call wrapper to compute tendencies in vector invariant eq.,
|       |       |           output = [guDissip,gvDissip] and net advective (inc. Coriolis) tendency [gU,gV] (via common block DYNVARS.h)
|       |       |
|       |       |START COMPUTING TENDENCY FROM DISSIPATION (EXPLICIT PARTS = harmonic + biharmonic + side drag + bottom drag)
|       |       |Do j=1-OLy,sNy+OLy Initialise local array for explicit dissipation tendency to 0
|       |       | Do i=i-OLx,sNx+OLx
|       |       |     guDiss(i,j) = 0.
|       |       |     gvDiss(i,j) = 0.
|       |       | ENDDO
|       |       |ENDDO
|       |       |
|       |       |--- mom_calc_ke KE at tracer point, needed for advective tendency, output=KE
|       |       |--- mom_calc_relvort3 k̂ · (∇ × u), needed for Leith eddy viscosity, output=vort3, without dynamic BC applied
|       |       |--- mom_calc_hdiv (∇_h · u), needed for Leith eddy viscosity, output=hDiv
|       |       |
|       |       |
|       |       |BEFORE COMPUTING DISSIPATION, UPDATE BACKGROUND VISCOSITY TO INCLUDE EDDY VISCOSITY BASED ON LOCAL FLOW STRUCTURE
|       |       |IF (useVariableVisc) THEN tension & strain also computed here but only relevant if using Smagorinsky closure
|       |       |                                   (not in ASTE_R1)
|       |       |--- mom_calc_visc add to preset viscosity an eddy viscosity based on gradient of vorticity and divergence (Leith)
|       |       |                   If Smagorinsky is used, eddy viscosity is based on strain and tension instead
|       |       |                   output = updated harmonic and biharmonic coeffs. at corner and centre points. See MITgcm manual section 2.21.1
|       |       |ENDIF
|       |       |
|       |       |IF (useBiharmonicVisc)
|       |       |--- mom_calc_del2uv Harmonic parts (i.e., D, ζ terms, not D*, ζ* terms) of Eq. (12) output=[del2u,del2v] (NB: ≠ ∇²u)
|       |       |--- mom_calc_hdiv 2nd call computes ∇· [Eq. (12) without the D*, ζ* terms]. Gives ∇ · (∇²u), output=dStar (Eq.(14))
|       |       |--- mom_calc_relvort3 2nd call computes k̂ · (∇× [Eq. (12) without the D*, ζ* terms]). Gives ∇²ζ3, output=zStar (Eq.(15))
|       |       |ENDIF
|       |       |
|       |       |If (.NOT. useStrainTensionVisc) THEN using vorticity and divergence formulation
|       |       |--- mom_vi_hdissip compute harmonic and biharmonic dissip. tendency as given by Eq. (12), from D, ζ, D*, ζ*
|       |       |     |                 output = updated dissip. tendencies [guDiss,gvDiss], but still missing side and bottom drag
|       |       |     |                 We have added a call to store the laplacian and biharmonic contributions separately
|       |       |     |CALL DIAGNOSTICS_FILL( uDiss_lap,'Um_Diss2',k,1,2,bi,bj,myThid)
|       |       |     |CALL DIAGNOSTICS_FILL( vDiss_lap,'Vm_Diss2',k,1,2,bi,bj,myThid)
|       |       |     |CALL DIAGNOSTICS_FILL( uDiss_bih,'Um_Diss4',k,1,2,bi,bj,myThid)
|       |       |     |CALL DIAGNOSTICS_FILL( vDiss_bih,'Vm_Diss4',k,1,2,bi,bj,myThid)
|       |       |     o
|       |       | ENDIF
|       |       |
|       |       | If (no_slip_sides) THEN
|       |       |--- mom_u_sidedrag u-mom tendency from no slip condition on viscous stress. Output = body force vF
|       |       |     |CALL DIAGNOSTICS_FILL( uDragTerms,'USidDrag',k,1,2,bi,bj,myThid)
|       |       |     o
|       |       | Do j=jMin,jMax update u-mom diss. tendency after exiting side drag S/R
|       |       |  Do i=iMin,iMax
```

```
|       |       |      guDiss(i,j) = guDiss(i,j) + vF(i,j)
|       |       |  ENDDO
|       |       | ENDDO
|       |       |ENDIF
|       |       |
|       |       |If (bottomDragTerms) THEN
|       |       |--- mom_u_botdrag  Apply no slip at bottom AND add additional friction as a function of the bottom speed or (in ASTE_R1) squared speed.
|       |       |      |                Output = body force vF
|       |       |      |CALL DIAGNOSTICS_FILL( uDragTerms,'UBotDrag',k,1,2,bi,bj,myThid)
|       |       |      o
|       |       | Do j=jMin,jMax  update u-mom diss. tendency after exiting bottom drag S/R
|       |       |  Do i=iMin,iMax
|       |       |      guDiss(i,j) = guDiss(i,j) + vF(i,j)
|       |       |  ENDDO
|       |       | ENDDO
|       |       |ENDIF
|       |       |
|       |       |NB for shelf ice configs (not in ASTE_R1) another u-mom tendency will be added here from shelf ice drag and an associated diagnostic should be written
|       |       |
|       |       |If (no_slip_sides) THEN
|       |       |--- mom_v_sidedrag  v-mom tendency from no slip condition on viscous stress. Output = body force vF
|       |       |      |CALL DIAGNOSTICS_FILL( vDragTerms,'VSidDrag',k,1,2,bi,bj,myThid)
|       |       |      o
|       |       | Do j=jMin,jMax  update v-mom diss. tendency after exiting side drag S/R
|       |       |  Do i=iMin,iMax
|       |       |      gvDiss(i,j) = gvDiss(i,j) + vF(i,j)
|       |       |  ENDDO
|       |       | ENDDO
|       |       |ENDIF
|       |       |
|       |       |If (bottomDragTerms) THEN
|       |       |--- mom_v_botdrag
|       |       |      |CALL DIAGNOSTICS_FILL( vDragTerms,'VBotDrag',k,1,2,bi,bj,myThid)
|       |       |      o
|       |       | Do j=jMin,jMax  update v-mom diss. tendency after exiting bottom drag S/R
|       |       |  Do i=iMin,iMax
|       |       |      gvDiss(i,j) = gvDiss(i,j) + vF(i,j)
|       |       |  ENDDO
|       |       | ENDDO
|       |       |ENDIF
|       |       |
|       |       |NB for shelf ice configs (not in ASTE_R1) another v-mom tendency will be added here from shelf ice drag and an associated diagnostic should be written
|       |       |
|       |       |FINISHED COMPUTING TENDENCY FROM DISSIPATION (EXPLICIT PARTS = harmonic + biharmonic + side drag + bottom drag)
|       |       |
|       |       |START COMPUTING TENDENCY FROM ADVECTION (Coriolis + 3 parts of inertia)
|       |       |If (useCoriolis .AND. .NOT. useCDscheme .AND. .NOT. useAbsVorticity) THEN
|       |       |---mom_vi_coriolis  Compute Coriolis acceleration [vf,-uf], output = [uCf,vCf]
```

```
|       |       | Do j=jMin,jMax  initialise u-mom and v-mom advective tendency with Coriolis tendency, arrays [gU,gV] are in common block DYNVARS.h
|       |       |  Do i=iMin,iMax
|       |       |     gU(i,j,k,bi,bj) = uCf(i,j)
|       |       |     gV(i,j,k,bi,bj) = vCf(i,j)
|       |       |  ENDDO
|       |       | ENDDO
|       |       |CALL DIAGNOSTICS_FILL( uCf,'Um_Cori',k,1,2,bi,bj,myThid)
|       |       |CALL DIAGNOSTICS_FILL( vCf,'Vm_Cori',k,1,2,bi,bj,myThid)
|       |       | ENDIF
|       |       |
|       |       | If ( momAdvection ) THEN
|       |       | If ( .NOT. (highOrderVorticity.OR.upwindVorticity.OR.useAbsVorticity) THEN
|       |       |---mom_vi_u_coriolis  inputs = (vFld,vort3) to compute u-mom forcing from vorticity advection: v\zeta_3 (see Eq. (9)), output = uCf
|       |       |  Do j=jMin,jMax  update u-mom advective tendency to include vorticity advection
|       |       |  Do i=iMin,iMax
|       |       |     gU(i,j,k,bi,bj) = gU(i,j,k,bi,bj) + uCf(i,j)
|       |       |  ENDDO
|       |       | ENDDO
|       |       |---mom_vi_v_coriolis  inputs = (uFld,vort3) to compute v-mom forcing from vorticity advection: -u\zeta_3 (see Eq. (9)), output = vCf
|       |       |  Do j=jMin,jMax  update v-mom advective tendency to include vorticity advection
|       |       |  Do i=iMin,iMax
|       |       |     gV(i,j,k,bi,bj) = gV(i,j,k,bi,bj) + vCf(i,j)
|       |       |  ENDDO
|       |       | ENDDO
|       |       |CALL DIAGNOSTICS_FILL( uCf,'Um_AdvZ3',k,1,2,bi,bj,myThid)
|       |       |CALL DIAGNOSTICS_FILL( vCf,'Vm_AdvZ3',k,1,2,bi,bj,myThid)
|       |       | ENDIF
|       |       |
|       |       | If ( .NOT. momImplVertAdv ) THEN
|       |       |---mom_vi_u_vertshear  inputs = (uVel,wVel) to compute u-mom tend. from vertical shears: -w\partial u/\partial r (see Eq. (10)), output = uCf
|       |       |  Do j=jMin,jMax  update u-mom advective tendency to include vertical shears term
|       |       |  Do i=iMin,iMax
|       |       |     gU(i,j,k,bi,bj) = gU(i,j,k,bi,bj) + uCf(i,j)
|       |       |  ENDDO
|       |       | ENDDO
|       |       |---mom_vi_v_vertshear  inputs = (vVel,wVel) to compute v-mom tend. from vertical shears: -w\partial v/\partial r (see Eq. (10)), output = vCf
|       |       |  Do j=jMin,jMax  update v-mom advective tendency to include vertical shears term
|       |       |  Do i=iMin,iMax
|       |       |     gV(i,j,k,bi,bj) = gV(i,j,k,bi,bj) + vCf(i,j)
|       |       |  ENDDO
|       |       | ENDDO
|       |       |CALL DIAGNOSTICS_FILL( uCf,'Um_AdvRe',k,1,2,bi,bj,myThid)
|       |       |CALL DIAGNOSTICS_FILL( vCf,'Vm_AdvRe',k,1,2,bi,bj,myThid)
|       |       | ENDIF
|       |       |
|       |       |---mom_vi_u_grad_ke  input = KE, to compute u-mom tend. from Bernoulli part of advection: \partial KE/\partial x (see Eq. (11)), output = uCf
|       |       |  Do j=jMin,jMax  update u-mom advective tendency to include Bernoulli term
|       |       |  Do i=iMin,iMax
```

```
|       |       |      gU(i,j,k,bi,bj) = gU(i,j,k,bi,bj) + uCf(i,j)
|       |       |     ENDDO
|       |       |    ENDDO
|       |       |---mom_vi_v_grad_ke  input = KE, to compute v-mom tend. from Bernoulli part of advection: ∂KE/∂y (see Eq. (11)), output = vCf
|       |       |    Do j=jMin,jMax  update v-mom advective tendency to include Bernoulli term
|       |       |     Do i=iMin,iMax
|       |       |      gV(i,j,k,bi,bj) = gV(i,j,k,bi,bj) + vCf(i,j)
|       |       |     ENDDO
|       |       |    ENDDO
|       |       |   CALL DIAGNOSTICS_FILL( uCf,'Um_dKEdx',k,1,2,bi,bj,myThid)
|       |       |   CALL DIAGNOSTICS_FILL( vCf,'Vm_dKEdy',k,1,2,bi,bj,myThid)
|       |       |   ENDIF /momAdvection
|       |       |
|       |       |   Finally, there's a call to compute non-traditional (cosine) Coriolis component to add on if use3dCoriolis=.TRUE..
|       |       |   This contribution will be zero if hydrostatic i.e., including for ASTE_R1
|       |       |   If ( use3dCoriolis ) THEN
|       |       |---mom_u_coriolis_nh  input = wVel, to compute u-mom contrib. from horizontal comp of Earth's rotation, output = uCf
|       |       |    Do j=jMin,jMax  update u-mom advective tendency to include non-traditional Coriolis
|       |       |     Do i=iMin,iMax
|       |       |      gU(i,j,k,bi,bj) = gU(i,j,k,bi,bj) + uCf(i,j)
|       |       |     ENDDO
|       |       |    ENDDO
|       |       |   If ( usingCurvilinearGrid ) THEN
|       |       |---mom_v_coriolis_nh  input = wVel, to compute v-mom contrib. from horizontal comp of Earth's rotation, output = vCf
|       |       |    Do j=jMin,jMax  update v-mom advective tendency to include non-traditional Coriolis
|       |       |     Do i=iMin,iMax
|       |       |      gV(i,j,k,bi,bj) = gV(i,j,k,bi,bj) + vCf(i,j)
|       |       |     ENDDO
|       |       |    ENDDO
|       |       |   ENDIF /usingCurvilinearGrid
|       |       |   CALL DIAGNOSTICS_FILL( uCf,'Um_CorNL',k,1,2,bi,bj,myThid)
|       |       |   CALL DIAGNOSTICS_FILL( vCf,'Vm_CorNL',k,1,2,bi,bj,myThid)
|       |       |   ENDIF /use3dCoriolis
|       |       |
|       |       |   Also, if non-hydrostatic dynamics are included, there's an additional contribution to advection here.
|       |       |
|       |       |   FINISHED COMPUTING TENDENCY FROM ADVECTION
|       |       |    Do j=jMin,jMax  apply kinematic BC
|       |       |     Do i=iMin,iMax
|       |       |      gU(i,j,k,bi,bj) = gU(i,j,k,bi,bj)*_maskW*(i,j,k,bi,bj)
|       |       |      gV(i,j,k,bi,bj) = gV(i,j,k,bi,bj)*_maskS*(i,j,k,bi,bj)
|       |       |     ENDDO
|       |       |    ENDDO
|       |       |
|       |       |   CALL DIAGNOSTICS_FILL( gU,'Um_Advec',k,1,2,bi,bj,myThid) Save net advec. tendency
|       |       |   CALL DIAGNOSTICS_FILL( gV,'Vm_Advec',k,1,2,bi,bj,myThid)
|       |       o
|       |
```

```
|       |DISSIPATIVE TENDENCY HAS BEEN COMPUTED AND OUTPUT IN ARRAYS [guDissip,gvDissip].
|       |EXPLICIT PGF IS IN ARRAYS [dPhiHydX,dPhiHydY] AND [phiSurfX,phiSurfY].
|       |ON LEAVING S/R mom_vecinv.F FORCING ARRAYS [gU,gV] (in DYNVARS.h) CONTAIN ONLY ADVECTIVE (INC. CORIOLIS) TENDENCY.
|       |MOVING ON TO COMPUTE REMAINING (EXF) TENDENCY AND ACCELERATE THE FLOW
|       |
|       |If using Smagorinsky (3D turbulent cascade) viscosity (not in ASTE_R1), an extra contribution is added to the dissipative tendency here.
|       |
|       |--- timestep  Compute remaining tendency from external forcing and accelerate the flow
|       |       |          input guDissip, gvDissip, dPhiHydX, dPhyHydY, phiSurfXm phiSurfY, and advective tendency [gU,gV] via include DYNVARS.h
|       |       |
|       |       |
|       |       |DO j= 1-oLy,sNy+OLy  Initialise external forcing tendency
|       |       | DO i=1-OLx,sNx+OLx
|       |       |  guExt(i,j)=0._d0
|       |       |  gvExt(i,j)=0._d0
|       |       | ENDDO
|       |       |ENDDO
|       |       |
|       |       | If ( momForcing ) THEN
|       |       |--- apply_forcing_u  Compute u-mom tendency from external forcing (just wind in ASTE_R1), output = guExt
|       |       |       |                  Here there are many S/R calls if one is using sophisticated atmospheres.
|       |       |       |                  In ASTE_R1, this forcing is simply computed from surfaceForcingU passed in common block SURFACE_FORCING.h
|       |       |       |CALL DIAGNOSTICS_FILL( gU_Wind,'Um_Wind',k,1,2,bi,bj,myThid)
|       |       |       |Other terms (none of which are used in ASTE_R1) can contribute to the external forcing tendency here.
|       |       |       |These are associated with (1) relaxation boundary conditions, (2) sponge open boundaries, and (3) prescribed GM streamfunction (I think)
|       |       |      o
|       |       |--- apply_forcing_v  Compute v-mom tendency from external forcing (just wind in ASTE_R1), output = gvExt.
|       |       |       |CALL DIAGNOSTICS_FILL( gV_Wind,'Vm_Wind',k,1,2,bi,bj,myThid)
|       |       |       |As above, other terms (none of which are used in ASTE_R1) can contribute to the external forcing tendency here.
|       |       |      o
|       |       |
|       |       | If ( momForcing .AND. momForcingOutAB.NE.1) THEN
|       |       |  Do j=jMin,jMax  update tendency (currently inertia + Coriolis) to include external forcing before doing AB
|       |       |   Do i=iMin,iMax
|       |       |      gU(i,j,k,bi,bj) = gU(i,j,k,bi,bj) + guExt(i,j)
|       |       |      gV(i,j,k,bi,bj) = gV(i,j,k,bi,bj) + gvExt(i,j)
|       |       |  ENDDO
|       |       |  ENDDO
|       |       | ENDIf
|       |       |
|       |       |--- adams_bashforth2  extrapolate forward in time using 2nd order AB
|       |       |     to avoid a mismatch between output [TOTUTEND,TOTVTEND] and sum of RHS tendencies, offsets due to stagger are stored as AB tendencies
|       |       |     output = [gu_AB,gv_AB]
|       |       |CALL DIAGNOSTICS_FILL(gu_AB,'AB_gU ',k,1,2,bi,bj,myThid)
|       |       |CALL DIAGNOSTICS_FILL(gu_AB,'AB_gU ',k,1,2,bi,bj,myThid)
|       |       |
|       |       | If ( .NOT. useCDscheme ) THEN
|       |       |  DO j = jMin,jMax  local copy of current tendency (inertia + Coriolis + external forcing)
```

```
|       |       |  DO i = iMin,iMax...old arrays will be overwritten with updated velocity
|       |       |     gUtmp(i,j) = gU(i,j,k,bi,bj)
|       |       |     gVtmp(i,j) = gV(i,j,k,bi,bj)
|       |       |  ENDDO
|       |       | ENDDO
|       |       |ENDIf
|       |       |
|       |       |If (momViscosity .AND. .NOT. momDissip_In_AB ) THEN
|       |       | DO j = jMin,jMax  update tendency (currently inertia + Coriolis + external forcing) to include dissipation)
|       |       |  DO i = iMin,iMax
|       |       |     gUtmp(i,j) = gUtmp(i,j) + guDissip(i,j)
|       |       |     gVtmp(i,j) = gVtmp(i,j) + gvDissip(i,j)
|       |       |  ENDDO
|       |       | ENDDO
|       |       |
|       |       | If using NH pressure (not in ASTE_R1), forcing is applied here and call to write associated tendency should be added here
|       |       |
|       |       | DO j = jMin,jMax  accelerate the flow by applying inertia + Coriolis + wind + dissipation forcing and total PGF
|       |       |  DO i = iMin,iMax  Note after this loop, [gU,gV] arrays contain velocity not acceleration
|       |       |   gU(i,j,k,bi,bj) = uVel(i,j,k,bi,bj) + deltaTMom*[gUtmp(i,j) -
|       |       |                           psFac*phiSurfX(i,j) - phxFac*dPhiHydX(i,j)
|       |       |                           ]*_maskW(i,j,k,bi,bj)
|       |       |  ENDDO
|       |       | ENDDO
|       |       | DO j = jMin,jMax  accelerate the flow by applying inertia + + Coriolis + wind + dissipation forcing and total PGF
|       |       |  DO i = iMin,iMax  Note after this loop, [gU,gV] arrays contain velocity not acceleration
|       |       |   gV(i,j,k,bi,bj) = vVel(i,j,k,bi,bj) + deltaTMom*[gVtmp(i,j) -
|       |       |                           psFac*phiSurfY(i,j) - phyFac*dPhiHydY(i,j)
|       |       |                           ]*_maskS(i,j,k,bi,bj)
|       |       |  ENDDO
|       |       | ENDDO
|       |       |
|       |       |If ( momViscosity) THEN
|       |       |CALL DIAGNOSTICS_FILL(guDissip,'Um_Diss ',k,1,2,bi,bj,myThid)
|       |       |CALL DIAGNOSTICS_FILL(gvDissip,'Vm_Diss ',k,1,2,bi,bj,myThid)
|       |       |ENDIF
|       |       |
|       |       |If ( momForcing ) THEN
|       |       |CALL DIAGNOSTICS_FILL(guExt,'Um_Ext ',k,1,2,bi,bj,myThid)
|       |       |CALL DIAGNOSTICS_FILL(guExt,'Vm_Ext ',k,1,2,bi,bj,myThid)
|       |       |ENDIF
|       |      o
|       | ENDDO  end of dynamics k loop (1:Nr)
|       |
|       |ALMOST FINISHED...NEED ONLY TO ADD FORCING FROM IMPLICIT DISSIPATION AND CORRECT FOR DIVERGENCE
|       | NB: after leaving S/R timestep, [gU,gV] arrays hold updated velocity NOT acceleration.
|       |
|       |If ( momImplVertAdv ) THEN  FALSE in ASTE (and as default) so we do not call the following S/R
```

```
|      |--- mom_u_implicit_r.F  apply forcing from implicit vertical advection and viscosity using kappaRU computed way earlier in calc_viscosity.F
|      |--- mom_v_implicit_r.F  apply forcing from implicit vertical advection and viscosity using kappaRV computed way earlier in calc_viscosity.F
|      |ENDIF
|      |
|      |If ( implicitViscosity ) THEN  TRUE in ASTE (default = false) so we do call the following:
|      |--- impldiff.F  apply forcing from implicit vertical viscosity (= diffusivity) using [kappaRU,kappaRV] computed way earlier in calc_viscosity.F
|      |      |CALL DIAGNOSTICS_FILL(df,'VISrI_Um',k,1,2,bi,bj,myThid)
|      |      |CALL DIAGNOSTICS_FILL(df,'VISrI_Vm',k,1,2,bi,bj,myThid)
|      |      o
|      |ENDIF
|      |
|     o
|
|---solve_for_pressure  solve elliptic eq. for p and update free surface displacement
|
|---momentum_correction_step  correct divergence in flow field with sfc pressure term from updated (just) surface displacement
|      |       This is the final modification to the velocity (excluding obcs points). Here's where I output the sfc PGF
|      |CALL DIAGNOSTICS_FILL(gU_eta,'Um_dPsdx',1,1,2,bi,bj,myThid)
|      |CALL DIAGNOSTICS_FILL(gV_eta,'Vm_dPsdy',1,1,2,bi,bj,myThid)
|     o
|
| FINISHED TENDENCY CALCULATION, FINISHED VELOCITY UPDATE. ALL THAT REMAINS IS TO SAVE THE EULERIAN ACCELERATION
|
|---do_statevars_diags
|      |
|      |--- diagnostics_fill_state
|     o      |CALL DIAGNOSTICS_SCALE_FILL(uVel,tmpFac,1,TOTUTEND,0,Nr,-1,bi,bj,myThid)
|            |CALL DIAGNOSTICS_SCALE_FILL(vVel,tmpFac,1,TOTVTEND,0,Nr,-1,bi,bj,myThid)
|            o
|
|
|
o
```

7 Code/Namelist Availability

ASTE_R1 model configuration and inputs are available for download from:

$ASTE_PATH = https://web.corral.tacc.utexas.edu/OceanProjects/ASTE/.

Detailed instructions for re-running ASTE_R1 are given in the file **readme_rerun_ASTEr1.txt** included with the release. MITgcm manual section 3.5 gives complete build instructions. As noted here, the user should point to the two existing directories containing ASTE_R1 code modifications ($exptdir/code_ASTE_R1 and $exptdir/code_adv7_bypass_tamc), when generating the `Makefile` (from within $exptdir/build):

'-mods ../code_ASTE_R1 ../code_adv7_bypass_tamc'

To enable momentum diagnostic output for closed budget analyses, we have also made our additional code modifications available for download from Corral:

$ASTE_PATH/code_ASTE_R1_mombudg

These should be added to the experiment directory. The -mods command line option above should then be edited to:

'-mods ../code_ASTE_R1 ../code_adv7_bypass_tamc ../code_ASTE_R1_mombudg'

You can also download:

$ASTE_PATH/NAMELISTS_MOMBUDG/data.diagnostics

to add to your $exptdir/NAMELISTS_ASTE_R1 directory, for outputting all momentum diagnostics described in Table 2 at double precision. Note that `frequency` is set to 1 day (86400) for all diagnostics, as used for producing figures for the example shown below. You will probably want to increase this for long runs!

8 Example Momentum Budget Analysis

The diagnostic output described above permits inspection of individual contributions to a closed momentum budget in ASTE_R1 following Eqs. (20) and (21). To demonstrate this, we plot terms contributing to the u-momentum tendency at model levels 1 (z = 5 m) and 20 (z = 300 m). All terms are averages over day 10 of the integration. We also illustrate our additional decomposition of both the advective (Fig. 5) and dissipative tendencies (Fig. 6), as described in Eqs. (22) and (23), respectively. All figures are plotted with logarithmic intervals, to allow the relative amplitudes of distinct momentum tendencies to be easily compared and confidently demonstrate closure. Momentum closure is not achieved at the open boundaries of ASTE_R1 (in the North Pacific and South Atlantic) and these cells are masked. Finally, we highlight that the data release was accompanied by useful routines for reading and regridding ASTE_R1 fields. The reader is referred to Nguyen et al. [2021a] for description of these tools and how to obtain them.

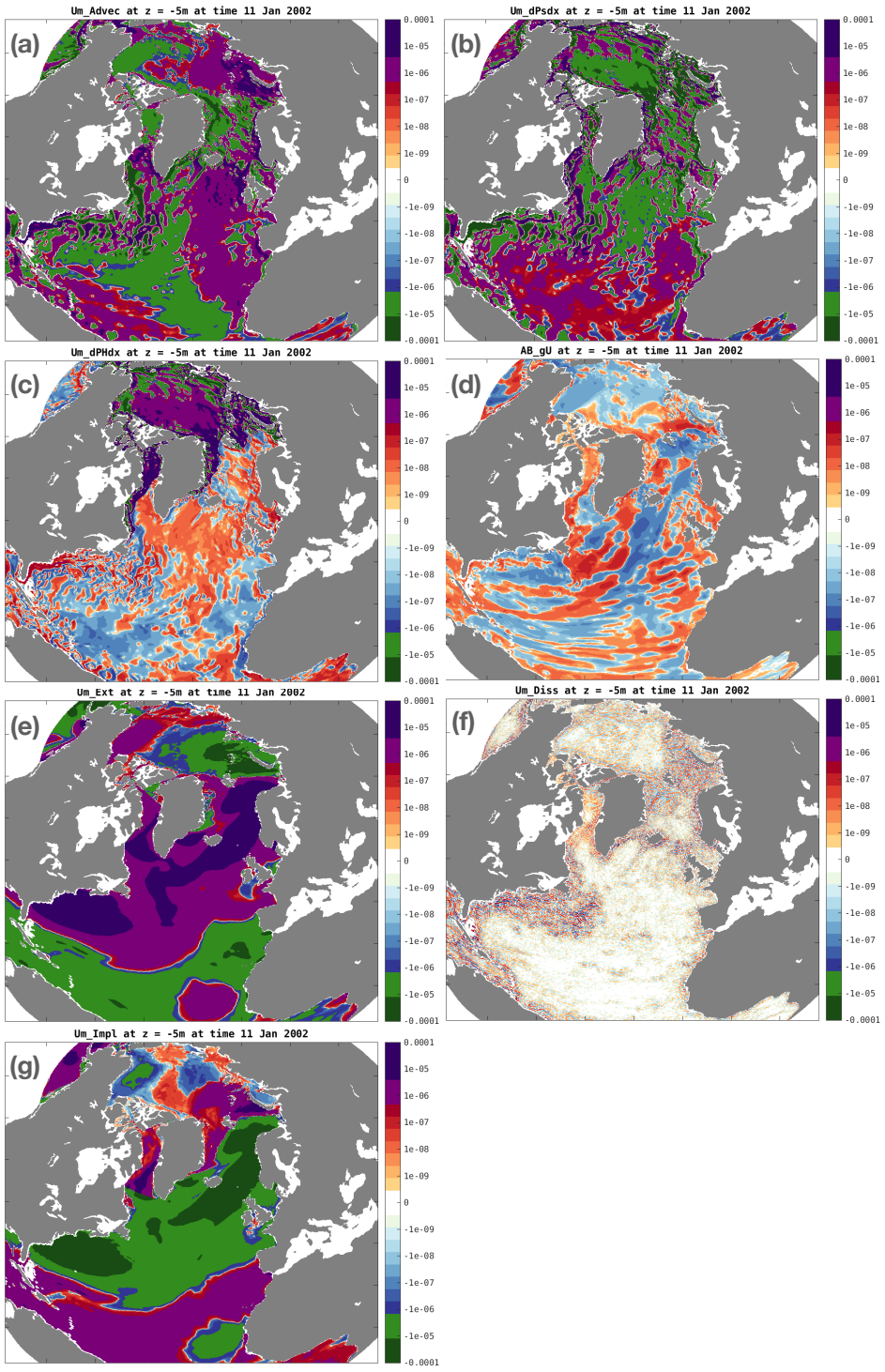

Figure 2: u-momentum tendency terms (ms^{-2}) at model level 1, averaged on 11 Jan 2002 of the 1/3° ASTE_R1 solution. These contributions from (a) `Um_Advec` = advection, (b) `Um_dPsdx` = surface displacement PGF, (c) `Um_dPHdx` = hydrostatic PGF, (d) `AB_gU` = Adams-Bashforth timestepping, (e) `Um_Ext` = wind forcing, (f) `Um_Diss` = dissipation, and (g) `Um_Impl` = implicit viscosity, combine to equal `TOTUTEND` = the Eulerian tendency of the zonal velocity to within numerical precision (see Fig. 4A1-A3).

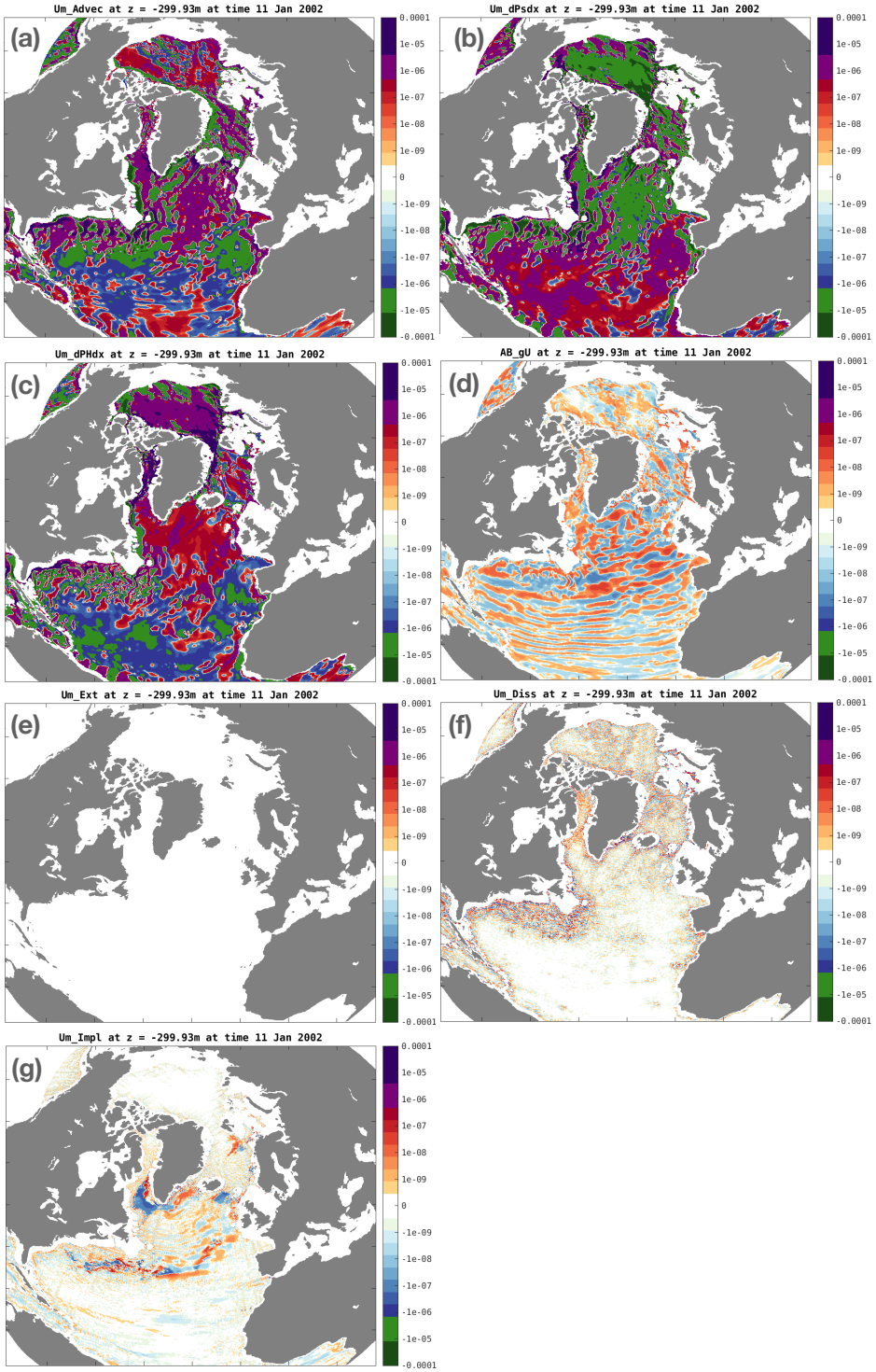

Figure 3: As for Fig. 2 but showing terms contributing to the u-momentum tendency at level 20 (depth = 299.93 m). Their sum explains the Eulerian tendency of the zonal velocity at this depth level (see Fig. 4B1-B3). In this configuration, only wind forcing contributes to `[Um_Ext,Vm_Ext]`, so these terms are zero for all depths but the surface level (panel e). Note that the surface PGF projects onto all depth levels and is the same here (panel b) as in Fig. 2b.

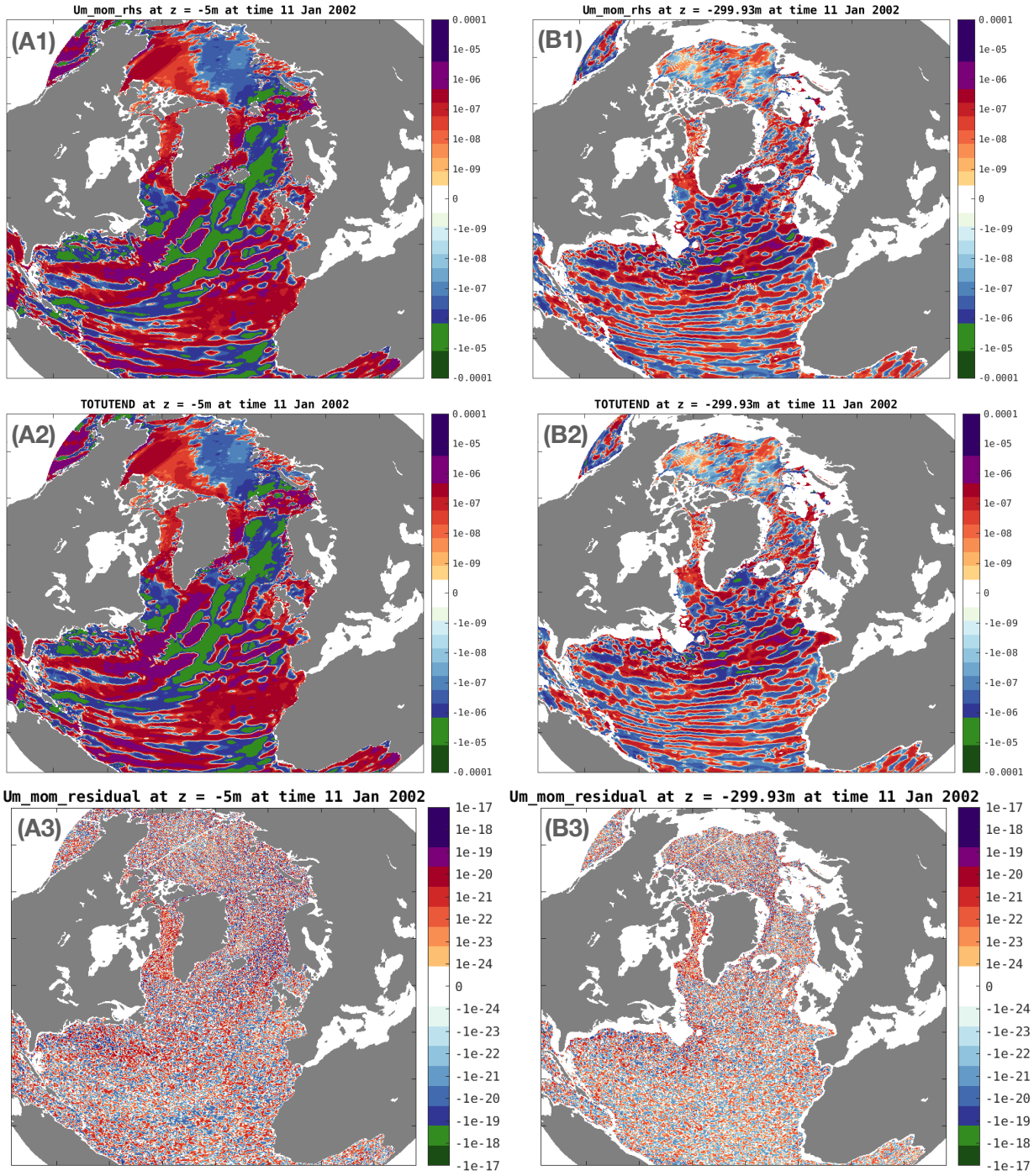

Figure 4: u-momentum closure at (A1-A3) level 1 (depth = 5 m) and (B1-B3) level 20 (depth = 299.93 m). Closure is determined by subtracting (A1,B1) the sum of all RHS tendencies as shown in Figs. 2 & 3 from (A2,B2) `TOTUTEND` = the Eulerian tendency of the zonal velocity. The residual is shown in (A3,B3) to be 14 orders of magnitude smaller than `TOTUTEND`. Note a different colorbar is used for the residuals.

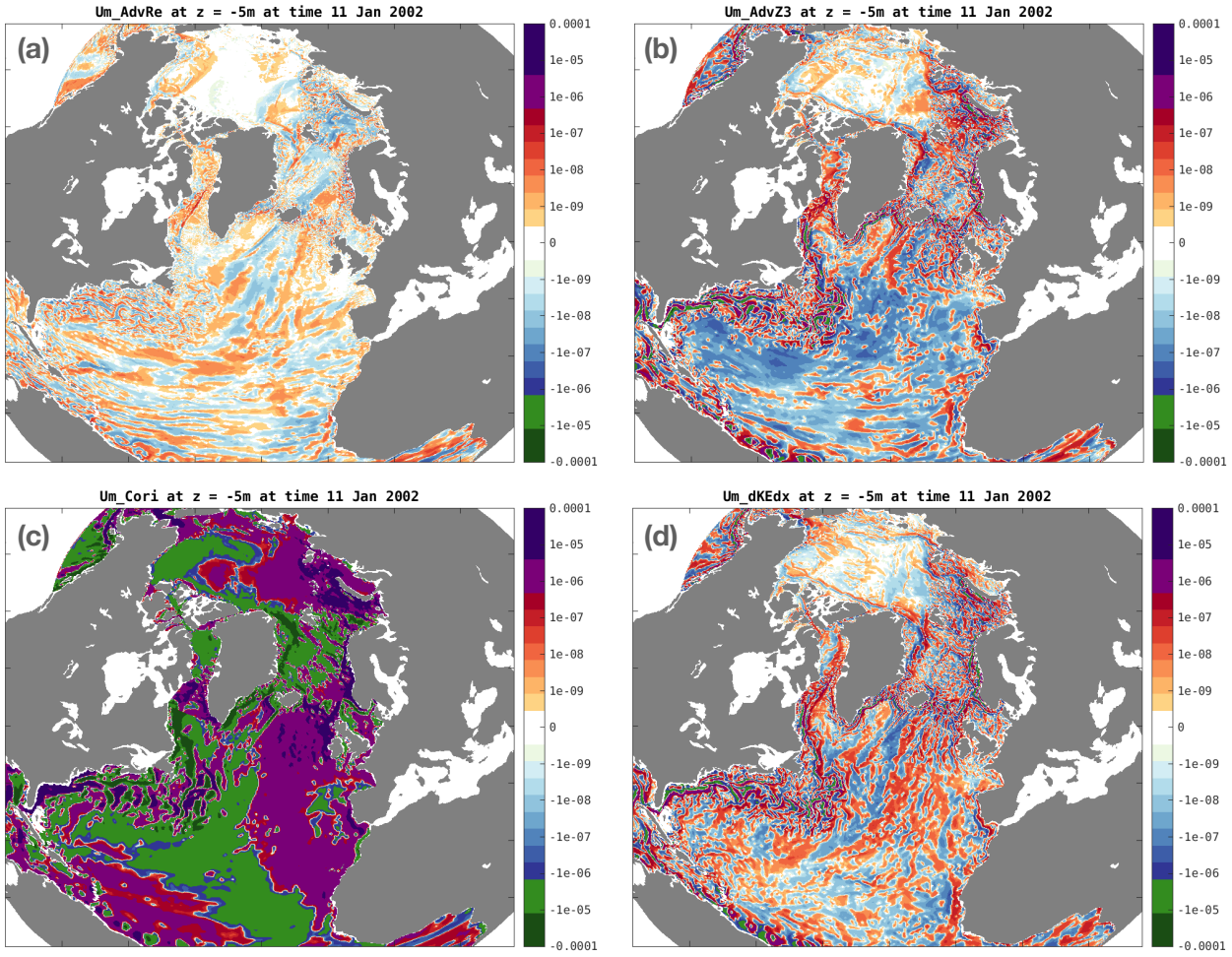

Figure 5: Decomposition of the u-momentum tendency from advection (Um_Advec) shown in Fig. 2 (z = 5 m) into separate contributions (a) `Um_AdvRe`, (b) `Um_AdvZ3`, (c) `Um_Cori`, and (d) `Um_dKEdx`, as given in Eq. (22).

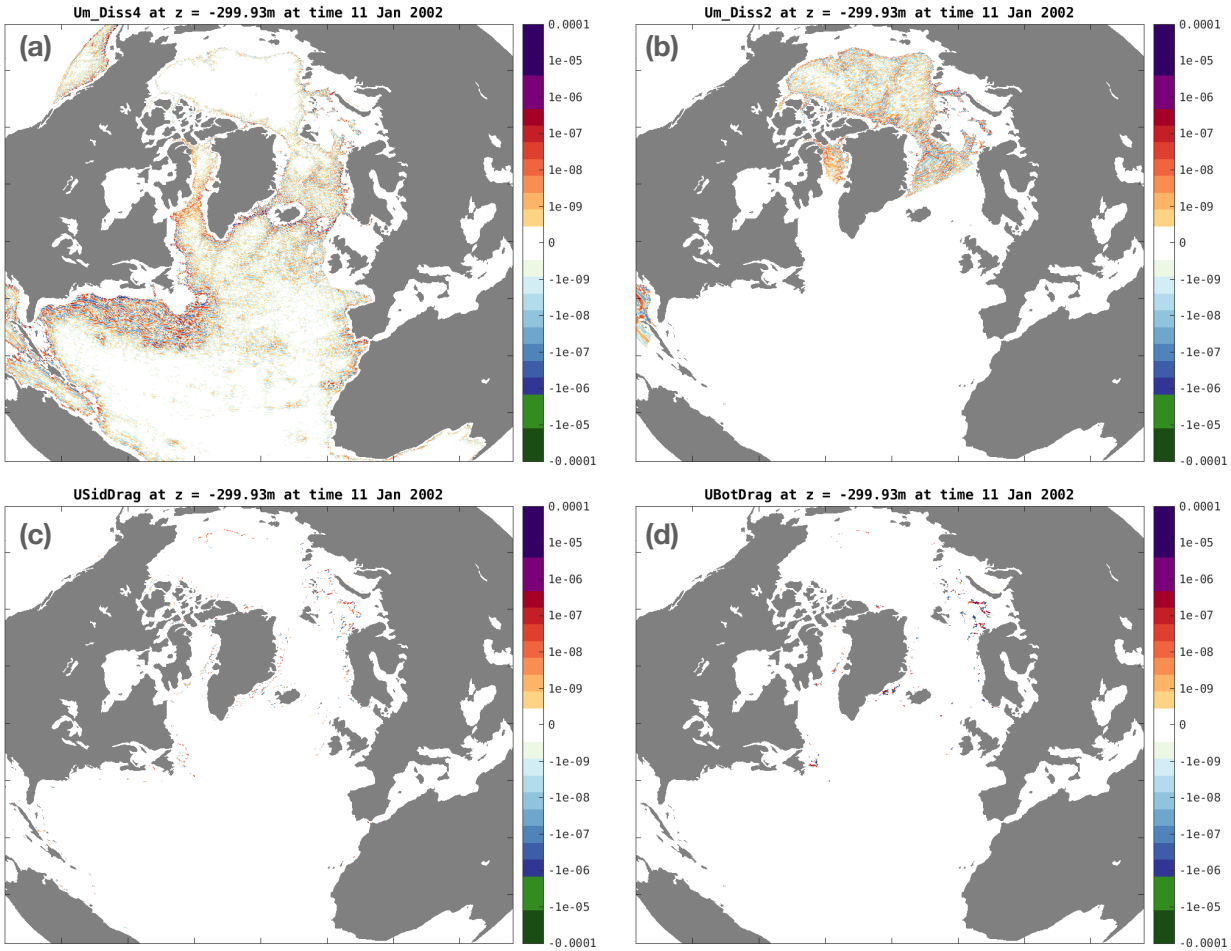

Figure 6: Decomposition of the u-momentum tendency from explicit dissipation (Um_Diss) shown in Fig. 3 (z = 299.93 m) into separate contributions (a) `Um_Diss4`, (b) `Um_Diss2`, (c) `USidDrag`, and (d) `UBotDrag`, as given in Eq. (23).

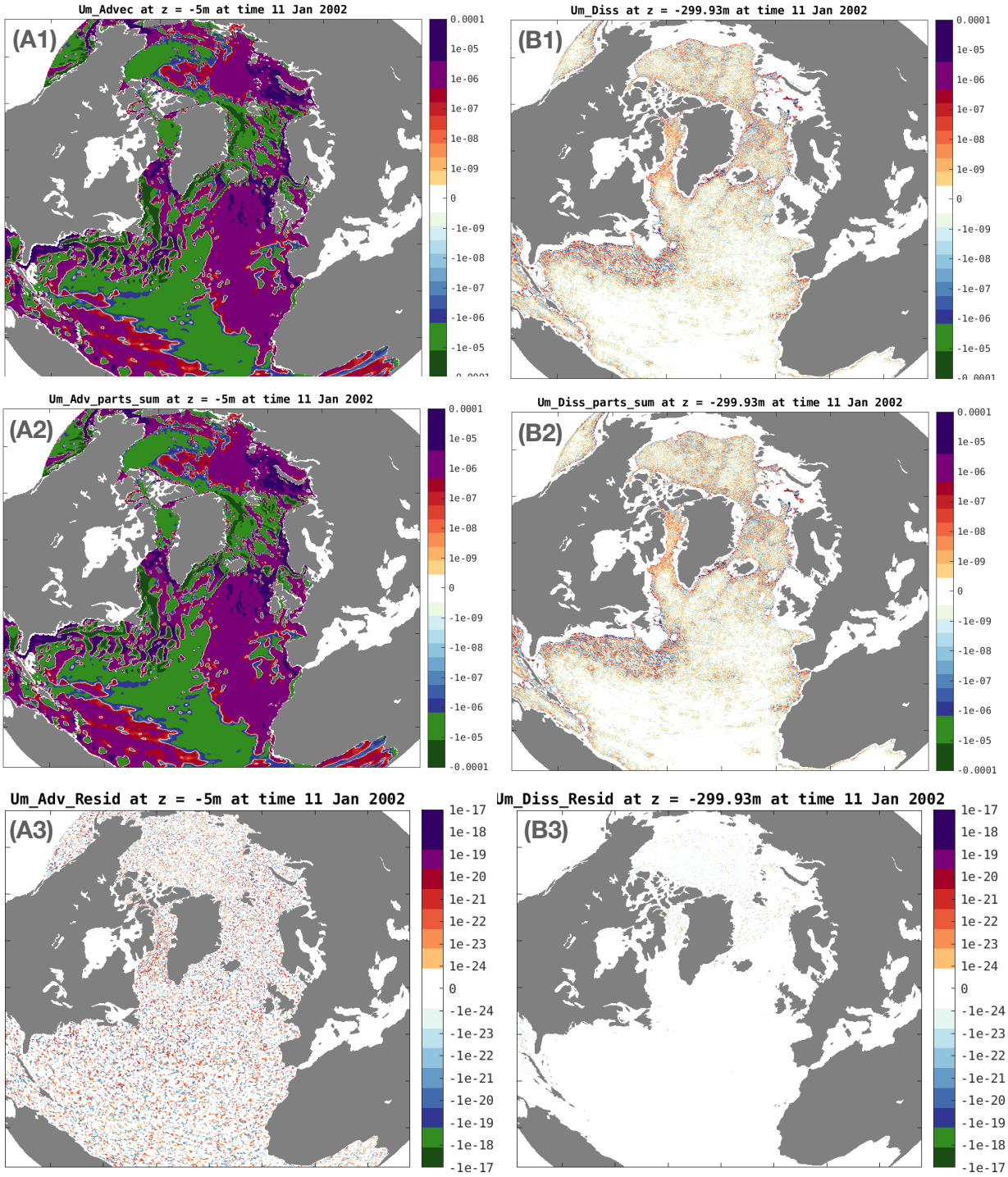

Figure 7: Confirmation the the tendencies for advection and explicit dissipation are accurately decomposed using Eqs. (22) & (23). The total tendencies from (A1) `Um_Advec` advection and (B1) `Um_Diss` explicit dissipation are equal to (A2,B2) the sum of their parts, shown in Figs. 5 and 6, respectively. The residuals are shown in (A3,B3) to be 16 orders of magnitude smaller than `Um_Advec` and `Um_Diss`. Note a different colorbar is used for the residuals.

References

Adcroft, A. and J.-M. Campin, 2004: Rescaled height coordinates for accurate representation of free-surface flows in ocean circulation models. *Ocean Modelling*, **7**, 269–284.

Adcroft, A., J.-M. Campin, S. Dutkiewicz, C. Evangelinos, D. Ferreira, G. Forget, B. Fox-Kemper, P. Heimbach, C. Hill, E. Hill, H. Hill, O. Jahn, M. Losch, J. Marshall, G. Maze, D. Menemenlis, and A. Molod, 2018: Mitgcm user manual. Tech. rep., MIT Department of EAPS, 77 Massachusetts Ave, Cambridge, MA.

Forget, G., J.-M. Campin, P. Heimbach, C. N. Hill, R. M. Ponte, and C. Wunsch, 2015: ECCO version 4: an integrated framework for non-linear inverse modeling and global ocean state estimation. *Geoscientific Model Development*, **8(10)**, 3071–3104.

Fox-Kemper, B. and D. Menemenlis, 2008: *Can Large Eddy Simulation Techniques Improve Mesoscale Rich Ocean Models?*, American Geophysical Union (AGU). ISBN 9781118666432, pp. 319–337.

Heimbach, P., I. Fukumori, C. Hill, R. Ponte, D. Stammer, J.-M. Campin, B. Cornuelle, I. Fenty, G. Forget, A. Koehl, M. Mazloff, D. Menemenlis, A. Nguyen, C. Piecuch, D. Trossman, A. Verdy, O. Wang, and H. Zhang, 2019: Putting it all together: Adding value to the global ocean and climate observing system with complete self-consistent ocean state and parameter estimates. *Front. Mar. Sci.*.

Large, W., J. McWilliams, and S. Doney, 1994: Oceanic vertical mixing: A review and a model with nonlocal boundary layer parameterization. *Rev. Geophys.*, **32**, 363–403.

Leith, C., 1996: Stochastic models of chaotic systems. *Physica D*, **98**, 481–491.

Marshall, J., A. Adcroft, C. Hill, L. Perelman, and C. Heisey, 1997a: A finite-volume, incompressible Navier Stokes model for studies of the ocean on parallel computers. *J. Geophys. Res.*, **102(C3)**.

Marshall, J., C. Hill, L. Perelman, and A. Adcroft, 1997b: Hydrostatic, quasi-hydrostatic, and non-hydrostatic ocean modeling. *J. Geophys. Res.*, **102(C3)**, 5733–5752.

Nguyen, A., P. Heimbach, T. Smith, and H. Pillar, 2021a: ASTE Release 1 User Guide. Tech. rep., Oden Institute for Computational Engineering and Sciences, The University of Texas at Austin, Austin.

Nguyen, A., H. Pillar, V. Ocaña, A. Bigdeli, T. A. Smith, and P. Heimbach, 2021b: The Arctic Subpolar gyre sTate Estimate (ASTE): Description and assessment of a data-constrained, dynamically consistent ocean-sea ice estimate for 2002-2017. *J. Adv. Model. Earth Sys.*, 62.

Piecuch, C., 2017: A note on evaluating budgets in ECCO Version 4 Release 3. Tech. rep.

Stammer, D., M. Balmaseda, P. Heimbach, A. Köhl, and A. Weaver, 2016: Ocean data assimilation in support of climate applications: status and perspectives. *Ann. Rev. Mar. Sci*, **8**, 491–518.

Stammer, D., C. Wunsch, R. Giering, C. Eckert, P. Heimbach, J. Marotzke, A. Adcroft, C. N. Hill, and J. Marshall, 2002: Global ocean circulation during 1992–1997, estimated from ocean observations and a general circulation model. *J. Geophys. Res.*, **107(C9)**, 1–27.

Stewart, A. and P. Dellar, 2012: Cross-equatorial channel flow with zero potential vorticity under the complete coriolis force. *IMA Journal of Applied Mathematics*, **77**, 626–651.

Wunsch, C. and P. Heimbach, 2007: Practical global ocean state estimation. *Physica D*, **230(1)**, 197–208.