A Note on Practical Evaluation of Budgets in ECCO Version 4 Release 3

Christopher G. Piecuch (cpiecuch@aer.com)

June 16, 2017

Contents

1 Introduction

Estimating the Circulation and Climate of the Ocean Version 4 (ECCOv4) is a new-generation estimate of the global ocean and sea-ice state (Forget et al. 2015). The estimate constitutes a solution to an ocean general circulation model, constrained to most available ocean data (altimetry, floats, etc.) using advanced inverse techniques. The ocean model solution describes the full-depth, three-dimensional, time-evolving oceanic state, including its changing sea level, heat, and salt content, among other state variables. The new ECCOv4 Release 3 solution covers the 1992–2015 period. See Forget et al. (2015) for more details on the ECCOv4 framework.

An important aspect is that, while constrained to data, the state estimate retains physical consistency. In other words, the solution exactly obeys conservation laws encoded in the model, and there are no nonphysical sources or sinks of volume, heat, salt, etc. This feature of the state estimate solution facilitates meaningful analysis of property budgets on the model grid, allowing changes in sea level, heat, salt content, etc., to be attributed unambiguously to the underlying causal mechanisms. For a recent example of budget studies using ECCOv4, see Thompson et al. (2016) for heat content in the Indian Ocean.

1.1 Scope

In what follows, we provide a basic outline, giving practical guidance for evaluating property budgets offline using available monthly ECCOv4 model diagnostic output. Importantly, note that methods described here are intended for analysis of model output on its native spatial grid (see sections 2 and *A*.1). Such methods are not appropriate for analysis of spatially interpolated model diagnostics, which are provided for convenience but not usable to evaluated budgets.

We emphasize that the (continuous and discretized) forms of the conservation equations presented here reflect the particular model configuration choices employed in ECCOv4 and described below in section 3. Therefore, the methods given below may not be appropriate for closing budgets under different model configuration choices. For example, a separate memo (Heat_Salt_Budget_MITgcm.pdf) discusses budgets for an earlier setup model setup (i.e., using different choices for the free surface condition and vertical coordinate).

2 Obtaining ECCOv4 Release 3

The ECCOv4 Release 3 solution can be downloaded from the ECCO server. The solution is in the form of monthly diagnostics, including basic ocean state variables (temperatures, salinities, velocities, etc.), surface forcing fields (e.g., wind stresses, heat fluxes), as well as other quantities needed for more advanced calculations and applications (advection and diffusion of temperature and salinity, bolus transport streamfunction, etc.). Specific model diagnostics needed for closing budgets for heat, salt, volume, etc., can be downloaded from the following two subdirectories—

- `/nctiles_monthly/`
- `/nctiles_monthly_snapshots/`

Note that model diagnostics can have large file sizes. For example, the full monthly potential temperature solution (`THETA`) is $\sim$ 10 GB.

Output diagnostics are provided in the form of NetCDF tiles, or `nctiles`. For a particular state variable (e.g., salinity, temperature, velocity), there are 13 such `nctiles`, each holding a horizontal "tile" of the full state estimate solution. The full solution is thus reconstituted by concatenating the `nctiles` together. The rationale for using this form of diagnostics is discussed in Appendix C of Forget et al. (2015).

3 Model Configuration

In the sections that follow, we introduce the conservation equations (budgets) used in ECCOv4, and how these budgets can be evaluated using model output in the context of offline analyses. However, the reader should note that the nature of the tracer conservation equations and surface boundary conditions used in ocean models can be sensitive to the details of model configuration. Thus, it is necessary first to discuss some details of the ECCOv4 model setup. Here we provide a brief outline. More detailed discussion is found in Section 3 of Forget et al. (2015).

The ECCOv4 state estimates are solutions to the MIT general circulation model, or `MITgcm` (Marshall et al. 1997). The particular configuration solves the primitive equations for the case of a Boussinesq, hydrostatic ocean. The model uses a nonlinear free surface and real freshwater exchanges. The model also uses a staggered time step, a vector-invariant form of the momentum

Parameter choice	Explanation
`implicitDiffusion`=.TRUE.,	Implicit vertical diffusion
`useRealFreshWaterFlux`=.TRUE.,	Real surface freshwater exchange
`select_rStar`=2,	Choice of rescaled vertical coordinate
`nonlinFreeSurf`=4,	Choice of nonlinear free surface
`implicitFreeSurface`=.TRUE.,	Implicit free surface
`exactConserv`=.TRUE.,	Exact conservation of global ocean volume
`tempAdvScheme`=30,	Multidimensional temperature advection
`saltAdvScheme`=30,	Multidimensional salt advection
`tempVertAdvScheme`=3,	Third-order vertical temperature advection
`saltVertAdvScheme`=3,	Third-order vertical salt advection
`tempImplVertAdv`=.TRUE.,	Implicit vertical temperature advection
`saltImplVertAdv`=.TRUE.,	Implicit vertical salt advection
`staggerTimeStep`=.TRUE.,	Staggered time step
`vectorInvariantMomentum`=.TRUE.,	Vector invariant momentum equations

Table 1: Model parameters (`PARM01`) in `MITgcm` configuration `data` file. See the `MITgcm` user manual for more general details.

equations, third-order Adams-Bashforth time-stepping (for advection and Coriolis terms in the momentum budget), direct space time (multidimensional) scheme for tracer advection, implicit tracer vertical advection and diffusion, and third-order vertical tracer advection. Key parameter choices related to this model configuration are given in Table 1.

The primitive equations are expressed in terms of a rescaled height coordinate,

$$z^* = \frac{z - \eta\left(x, y, t\right)}{H\left(x, y\right) + \eta\left(x, y, t\right)} H\left(x, y\right). \tag{1}$$

Here z is the unscaled vertical coordinate, η is surface height (at the air-sea or ice-sea interface), and H is ocean depth (Adcroft and Campin 2004). Note that the range of this rescaled height coordinate is $z^* \in [-H, 0]$. That is, the upper surface boundary in z^* is time invariant.

Diagnostic	Temporal	Description (Units)
`ETAN`	Snapshot	Surface height anomaly (m)
`oceFWflx`	Average	Net surface freshwater flux into the ocean (kg m^{-2} s^{-1})
`UVELMASS`	Average	Zonal mass-weighted component of velocity (m s^{-1})
`VVELMASS`	Average	Meridional mass-weighted component of velocity (m s^{-1})
`WVELMASS`	Average	Vertical mass-weighted component of velocity (m s^{-1})

Table 2: `MITgcm` diagnostics required to evaluate the vertically integrated volume budget.

4 Budgets Evaluated for Extensive Quantities

4.1 Volume Conservation

The equation for volume conservation (continuity) in the z^* reference frame is, in its continuous form (see equation 3 in Forget et al. 2015),

$$\underbrace{\frac{1}{H}\frac{\partial \eta}{\partial t}}_{G^{\eta,tot}} = \underbrace{-\nabla_{z^*}\left(s^*\mathbf{v}\right) - \frac{\partial w}{\partial z^*}}_{G^{\eta,conv}} + \underbrace{s^*\mathcal{F}}_{G^{\eta,forc}}, \tag{2}$$

Here $s^* = 1 + \eta/H$ is a scale factor, ∇_{z^*} and $\partial/\partial z^*$ are horizontal and vertical divergences in the z^* frame, respectively, $\mathbf{v} = (u, v)$ and w are the resolved horizontal and vertical velocities, respectively, and $\mathcal{F}$ is proportional to the volumetric freshwater flux forcing.

Taking into account time stepping scheme (Table 1), the discretized version of equation (2) diagnosed by the model and relating the updated state $(\eta^{n+1}, \mathbf{v}^{n+1}, w^{n+1})$ at time $t + \Delta t$ to the current state $(\eta^n, \mathbf{v}^n, w^n)$ at time t is (see equation B4 in Forget et al. 2015),

$$\frac{1}{H}\frac{\eta^{n+1} - \eta^n}{\Delta t} = -\nabla_{z^*}\left(s^{*n}\mathbf{v}^{n+1}\right) - \frac{\partial w^{n+1}}{\partial z^*} + s^{*n}\mathcal{F}^{n+1/2}, \tag{3}$$

where superscript denotes the time step corresponding to the particular variable.

Note that the forms of budgets (2) and (3) here for volume (and below for heat and salt) are that of total tendency ($G^{\eta,tot}$) on the left hand side (LHS) being balanced by the sum of ocean transport convergences ($G^{\eta,conv}$) and sea surface forcing ($G^{\eta,forc}$) on the right hand side (RHS). In the context of offline analysis, given a particular time period of interest, LHS tendency terms are evaluated based on *temporal snapshot* (or instantaneous) model output corresponding to the beginning and end of that time period, while RHS convergence and forcing terms are assessed using *temporal average* model output taken over the time interval.

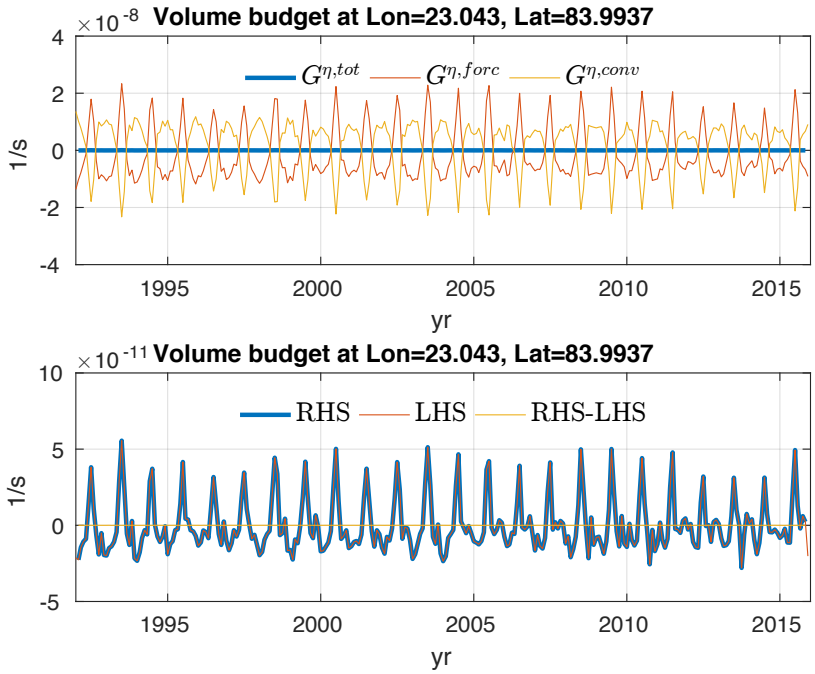

Figure 1: Volume budget for an arbitrary surface grid cell. Top panel shows the individual terms in the budget equation (2). Bottom panel shows the LHS, RHS, and difference between LHS and RHS terms in the budget. The good agreement between RHS and LHS (e.g., the ratio of the standard deviation of the residual to the standard deviation of the tendency here is $\mathcal{O}(10^{-3})$) demonstrates practical closure of the budget.

Table 2 lists the `MITgcm` diagnostics needed to evaluate the volume budget in offline analysis, while Algorithm (1) provides example pseudocode for closing the budget. (We give more specific Matlab code for evaluation of the volume budget in the Appendix.) An example volume budget for an arbitrary grid cell based on output in Table 2 is shown in Figure 1.

Note that there is very good agreement between the independently evaluated LHS tendency term and the sum of RHS convergence and forcing terms in Figure 1. More quantitatively, averaged over the global ocean surface in the first vertical layer ($k = 1$), the ratio of the standard deviation of the residual (LHS−RHS) to the tendency (LHS) in equation (2) using methods presented here is $\mathcal{O}(10^{-2})$. [Note that budget closure checks are routinely carried out as part of the "standard analysis" described in the supplement to Forget et al. (2015).] Similar results are seen for the cases of example heat and salt budgets shown below in Figures 2 and 3. This shows that, provided they are evaluated correctly, the RHS fluxes computed from monthly averages should match the LHS tendency calculated from instantaneous snapshots. (Indeed, the instantaneous snapshots are made available precisely to facilitate such offline consistency checks.)

Algorithm 1 : Evaluating the volume budget. Subscripts i, j, k denote spatial positions in x, y, z, respectively (except for the Kronecker delta $\delta_{a,b}$). The other terms are grid parameters: H is water column depth (`Depth`), h is grid cell relative thickness (`hFacC`), Δx is horizontal thickness of grid cell southern edge (`DXG`), Δy is horizontal thickness of grid cell western edge (`DYG`), and Δz is grid cell vertical thickness (`DRF`). For more general details on grid parameters, see MITgcm user manual Chapter 2. Grid parameters for ECCOv4 can be downloaded from the ECCO ftp server.

1.
for $t = t_1, t_2, \dots t_{T-1}, t_T$ **do** ▷ Loop over T time steps (months) t
  $F_{i,j} = \texttt{oceFWflx}\{t\}$ ▷ 2-D **average** freshwater flux over month t
  $U_{i,j,k} = \texttt{UVELMASS}\{t\}$ ▷ 3-D **average** zonal velocity over month t
  $V_{i,j,k} = \texttt{VVELMASS}\{t\}$ ▷ 3-D **average** meridional velocity over month t
  $W_{i,j,k} = \texttt{WVELMASS}\{t\}$ ▷ 3-D **average** vertical velocity over month t
  $N^{(0)}_{i,j} = \texttt{ETAN}\{t - \Delta t\}$ ▷ 2-D surface height **snapshot** at start of month t
  $N^{(f)}_{i,j} = \texttt{ETAN}\{t\}$ ▷ 2-D surface height **snapshot** at end of month t
  $\rho_0 = 1029$ ▷ Reference density (kg m^{-3})
  for $i = i_1, i_2, \dots i_{I-1}, i_I$ **do** ▷ Loop over I longitude cells i
    for $j = j_1, j_2, \dots j_{J-1}, j_J$ **do** ▷ Loop over J latitude cells j
      for $k = k_1, k_2, \dots k_{K-1}, k_K$ **do** ▷ Loop over K vertical cells k
        $G^{\eta,tot}_{i,j,k} = \left(N^{(f)}_{i,j} - N^{(0)}_{i,j}\right) / (H_{i,j}\Delta t)$
        $G^{\eta,forc}_{i,j,k} = \delta_{k,1} F_{i,j} / (\rho_0 h_{i,j,k} \Delta z_k)$
        $G^{\eta,convH}_{i,j,k} = \left[(U_{i,j,k} - U_{i+1,j,k})\,\Delta y_{i,j} + (V_{i,j,k} - V_{i,j+1,k})\,\Delta x_{i,j}\right] / (A_{i,j} h_{i,j,k})$
        $G^{\eta,convV}_{i,j,k} = \left[(1 - \delta_{k,K})\,W_{i,j,k+1} - (1 - \delta_{k,1})\,W_{i,j,k}\right] / (h_{i,j,k}\Delta z_k)$
        $G^{\eta,conv}_{i,j,k} = G^{\eta,convH}_{i,j,k} + G^{\eta,convV}_{i,j,k}$
      end for
    end for
  end for
end for

4.2 Heat Conservation

The heat conservation equation in z^* is (see equation 4 in Forget et al. 2015),

$$\underbrace{\frac{\partial\,(s^*\theta)}{\partial t}}_{G^{\theta,tot}} = \underbrace{-\nabla_{z^*}\,(s^*\theta \mathbf{v}_{res}) - \frac{\partial\,(\theta w_{res})}{\partial z^*}}_{G^{\theta,adv}} + \underbrace{s^*\mathcal{F}_\theta}_{G^{\theta,forc}} + \underbrace{s^* D_\theta}_{G^{\theta,diff}}. \tag{4}$$

Here θ is potential temperature, $\mathbf{v}_{res} = (u_{res}, v_{res})$ and w_{res} are the total horizontal and vertical velocities, respectively, $\mathcal{F}_\theta$ is total local forcing by surface heat exchanges, and D_θ symbolizes parameterized diffusive mixing processes. Total velocities $\mathbf{v}_{res}$ and w_{res} are sometimes called "residual mean" velocities. They contain both the resolved (Eulerian) flow field, as well as the "bolus" velocity, parameterizing unresolved eddy effects after Gent and McWilliams (1990). The diffusion term D_θ contains both diapycnal and isopycnal components, as well as turbulence in the mixed layer (Gaspar et al. 1990) and convection. Forcing $\mathcal{F}_\theta$ contains the latent, sensible, longwave, and shortwave components. Importantly, the shortwave radiative heat flux penetrates the water column vertically (see below).

Given the model time stepping, the discrete version of equation (4) relating the updated state $(\eta^{n+1}, \mathbf{v}^{n+1}, w^{n+1}, \theta^{n+3/2})$ at time $t + \Delta t$ to the current state $(\eta^n, \mathbf{v}^n, w^n, \theta^{n+1/2})$ at time t is (see equation B5 in Forget et al. 2015),

$$\frac{s^{*n+1}\theta^{n+3/2} - s^{*n}\theta^{n+1/2}}{\Delta t} = \mathcal{A}\left(\theta, \mathbf{u}^{n+1} + \mathbf{u}_b^{n+1}\right) + s^{*n}\left(\mathcal{F}_\theta^{n+1} + D_{\sigma,\theta}^{n+1/2} + D_{\perp,\theta}^{n+3/2}\right). \tag{5}$$

Here $\mathcal{A}\,()$ symbolizes the advection term, $\mathbf{u} = (u, v, w)$ the full three-dimensional velocity, $\mathbf{u}_b$ the bolus velocity, and subscripts σ and $\perp$ are the isopycnal and diapycnal components of the diffusion term D_θ, respectively.

Table 3 lists `MITgcm` diagnostics needed for evaluating monthly heat budgets with ECCOv4. Given the nature of the surface forcing term, we demonstrate evaluation of the heat budget in two parts. First, we deal with the total tendency term and ocean transport convergences. The operations sketched in Algorithm (2) for the tendency and transport terms in the heat budget are very similar to those given in Algorithm (1) for the analogous terms in the volume budget. (Note that we provide specific Matlab code for evaluation of the heat budget in the Appendix.)

Second, we tackle local surface heat flux forcing. To follow the relevant pseudocode outlined in Algorithm (3), one needs to understand how the `MITgcm` setup represents the local surface forcing term. In ECCOv4, shortwave radiation penetrates the water column vertically over the top 200 m as exponentially decaying Jerlov Type IA-2 water (Paulson and Simpson 1977),

$$Q^{sw}(z) = Q^{sw}(0)\frac{q_1 - q_2}{\Delta z}. \tag{6}$$

Algorithm 2 : Evaluating the tendency and transport terms in the heat budget. See Algorithm (1) caption for description of subscript indices, grid parameters, and other symbols. Readers interested in the details of these calculations as performed by the model are referred to the `MITgcm` subroutines `gad_calc_rhs.F` and `impldiff.F`.

1. **for** $t = t_1, t_2, \ldots t_{T-1}, t_T$ **do** ▷ Loop over T time steps (months) t
   - $U_{i,j,k} = \texttt{ADVx\_TH}\{t\}$ ▷ 3-D **average** zonal advection over month t
   - $V_{i,j,k} = \texttt{ADVy\_TH}\{t\}$ ▷ 3-D **average** meridional advection over month t
   - $W_{i,j,k} = \texttt{ADVr\_TH}\{t\}$ ▷ 3-D **average** vertical advection over month t
   - $\mathcal{U}_{i,j,k} = \texttt{DFxE\_TH}\{t\}$ ▷ 3-D **average** zonal diffusion over month t
   - $\mathcal{V}_{i,j,k} = \texttt{DFyE\_TH}\{t\}$ ▷ 3-D **average** meridional diffusion over month t
   - $\mathcal{W}^E_{i,j,k} = \texttt{DFyE\_TH}\{t\}$ ▷ 3-D **average** vertical diffusion (explicit) over month t
   - $\mathcal{W}^I_{i,j,k} = \texttt{DFyI\_TH}\{t\}$ ▷ 3-D **average** vertical diffusion (implicit) over month t
   - $N^{(0)}_{i,j} = \texttt{ETAN}\{t - \Delta t\}$ ▷ 2-D surface height **snapshot** at start of month t
   - $N^{(f)}_{i,j} = \texttt{ETAN}\{t\}$ ▷ 2-D surface height **snapshot** at end of month t
   - $T^{(0)}_{i,j,k} = \texttt{THETA}\{t - \Delta t\}$ ▷ 3-D temperature **snapshot** at start of month t
   - $T^{(f)}_{i,j,k} = \texttt{THETA}\{t\}$ ▷ 3-D temperature **snapshot** at end of month t
   - $v_{i,j,k} = h_{i,j,k} A_{i,j} \Delta z_k$ ▷ Grid volume
   - **for** $i = i_1, i_2, \ldots i_{I-1}, i_I$ **do** ▷ Loop over I longitude cells i
     - **for** $j = j_1, j_2, \ldots j_{J-1}, j_J$ **do** ▷ Loop over J latitude cells j
       - $s^{*(0)}_{i,j} = \left(1 + N^{(0)}_{i,j}/H_{i,j}\right)$
       - $s^{*(f)}_{i,j} = \left(1 + N^{(f)}_{i,j}/H_{i,j}\right)$
       - **for** $k = k_1, k_2, \ldots k_{K-1}, k_K$ **do** ▷ Loop over K vertical cells k
         - $G^{\theta,tot}_{i,j,k} = \left(T^{(f)}_{i,j,k} s^{*(f)}_{i,j} - T^{(0)}_{i,j,k} s^{*(0)}_{i,j}\right)/\Delta t$
         - $G^{\theta,advH}_{i,j,k} = \left(U_{i,j,k} - U_{i+1,j,k} + V_{i,j,k} - V_{i,j+1,k}\right)/v_{i,j,k}$
         - $G^{\theta,diffH}_{i,j,k} = \left(\mathcal{U}_{i,j,k} - \mathcal{U}_{i+1,j,k} + \mathcal{V}_{i,j,k} - \mathcal{V}_{i,j+1,k}\right)/v_{i,j,k}$
         - $G^{\theta,advV}_{i,j,k} = \left[(1 - \delta_{k,K}) W_{i,j,k+1} - W_{i,j,k}\right]/v_{i,j,k}$
         - $G^{\theta,diffV}_{i,j,k} = \left[(1 - \delta_{k,K})\left(\mathcal{W}^E_{i,j,k+1} + \mathcal{W}^I_{i,j,k+1}\right) - \mathcal{W}^E_{i,j,k} - \mathcal{W}^I_{i,j,k}\right]/v_{i,j,k}$
         - $G^{\theta,adv}_{i,j,k} = G^{\theta,advH}_{i,j,k} + G^{\theta,advV}_{i,j,k}$
         - $G^{\theta,diff}_{i,j,k} = G^{\theta,diffH}_{i,j,k} + G^{\theta,diffV}_{i,j,k}$
       - **end for**
     - **end for**
   - **end for**
1. **end for**

Diagnostic	Time	Description (Units)
ETAN	Snapshot	Surface height anomaly (m)
THETA	Snapshot	Potential temperature (°C)
TFLUX	Average	Total heat flux (W m^{-2})
oceQsw	Average	Net shortwave radiation (W m^{-2})
ADVr_TH	Average	Vertical advective flux of pot. temp. (°C m^3 s^{-1})
ADVx_TH	Average	Zonal advective flux of pot. temp. (°C m^3 s^{-1})
ADVy_TH	Average	Meridional advective flux of pot. temp. (°C m^3 s^{-1})
DFrI_TH	Average	Implicit vertical diffusive flux of pot. temp. (°C m^3 s^{-1})
DFrE_TH	Average	Explicit vertical diffusive flux of pot. temp. (°C m^3 s^{-1})
DFxE_TH	Average	Explicit zonal diffusive flux of pot. temp. (°C m^3 s^{-1})
DFyE_TH	Average	Explicit meridional diffusive flux of pot. temp. (°C m^3 s^{-1})

Table 3: MITgcm diagnostics required to evaluate the grid cell heat budget. In addition, to evaluate the globally averaged or deep ocean heat budget, the user needs the geothermal flux forcing file, as described below in section 4.2.1.

Here $Q^{sw}(z)$ is the shortwave radiation penetrating to depth z, Δz is the vertical thickness of the grid cell centered on z, and q_1 and q_2 are functions of depth given by,

$$q_i = 0.62 \exp\left(\frac{z_i}{0.6}\right) + (1 - 0.62) \exp\left(\frac{z_i}{20}\right) \ , \ i \in \{1, 2\} \, , \ z_i < 0 \tag{7}$$

where z_1 (z_2) is the depth of the "top" ("bottom") of the vertical grid cell. Thus, to properly evaluate the forcing term, the shortwave contribution (the oceQsw diagnostic) must be removed from the total flux (the TFLUX diagnostic) and redistributed in the vertical following equations (6) and (7).

Figure 2 shows an example heat budget at an arbitrary grid cell using output in Table 3. Averaged over the global ocean surface in the first vertical layer ($k = 1$), the ratio of the standard deviation of the residual (LHS−RHS) to the tendency (LHS) in equation (4) using methods presented here is $\mathcal{O}(10^{-5})$.

4.2.1 Geothermal Flux

A final detail with respect to the heat budget is that, for grid cells on the seafloor, $\mathcal{F}_\theta$ contains a contribution from geothermal flux (Piecuch et al. 2015). This detail is of particular relevance to readers interested in globally integrated or abyssal ocean heat budgets. This geothermal flux contribution is not accounted for in any of the standard model diagnostics provided as output. Rather, this term, which is time invariant, is provided in the input file geothermalFlux.bin

Algorithm 3 : Evaluating the forcing term in the heat budget. See Algorithm (1) caption for description of subscript indices, grid parameters, and other symbols. Readers more interested in the details of these calculations as performed by the model are referred to the `MITgcm` subroutines `external_forcing.F` and `swfrac.F`.

1. **for** $t = t_1, t_2, \ldots t_{T-1}, t_T$ **do** ▷ Loop over T time steps (months) t
   - $Q_{i,j} = \texttt{TFLUX}\,\{t\}$ ▷ 2-D **average** total heat flux over month t
   - $S_{i,j} = \texttt{oceQsw}\,\{t\}$ ▷ 2-D **average** shortwave radiation over month t
   - $\rho_0 = 1029$ ▷ Reference density (kg m^{-3})
   - $c_p = 3994$ ▷ Heat capacity (J kg^{-1} °C^{-1})
   - $R = 0.62$ ▷ Constant (cf. Paulson and Simpson 1977 Table 2)
   - $\zeta_1 = 0.6$ ▷ Constant (cf. Paulson and Simpson 1977 Table 2)
   - $\zeta_2 = 20$ ▷ Constant (cf. Paulson and Simpson 1977 Table 2)
   - **for** $k = k_1, k_2, \ldots k_{K-1}, k_K$ **do** ▷ Loop over K vertical cells k
     - **if** $0 > z_k > -200$ **then** ▷ If above 200 m depth
       - $q_{1,k} = R \exp\left(z_{1,k}/\zeta_1\right) + (1-R) \exp\left(z_{1,k}/\zeta_2\right)$
       - $q_{2,k} = R \exp\left(z_{2,k}/\zeta_1\right) + (1-R) \exp\left(z_{2,k}/\zeta_2\right)$
     - **else**
       - $q_{1,k} = 0$
       - $q_{2,k} = 0$
     - **end if**
   - **end for**
   - **for** $i = i_1, i_2, \ldots i_{I-1}, i_I$ **do** ▷ Loop over I longitude cells i
     - **for** $j = j_1, j_2, \ldots j_{J-1}, j_J$ **do** ▷ Loop over J latitude cells j
       - **for** $k = k_1, k_2, \ldots k_{K-1}, k_K$ **do** ▷ Loop over K vertical cells k
         - **if** $k = 1$ **then**
           - $G^{\theta,forc}_{i,j,k} = \langle Q_{i,j} - [1 - (q_{1,k} - q_{2,k})]\, S_{i,j} \rangle / \left(\rho_0 c_p h_{i,j,k} \Delta z_k\right)$
         - **else**
           - $G^{\theta,forc}_{i,j,k} = \left[(q_{1,k} - q_{2,k})\, S_{i,j}\right] / \left(\rho_0 c_p h_{i,j,k} \Delta z_k\right)$
         - **end if**
       - **end for**
     - **end for**
   - **end for**
2. **end for**

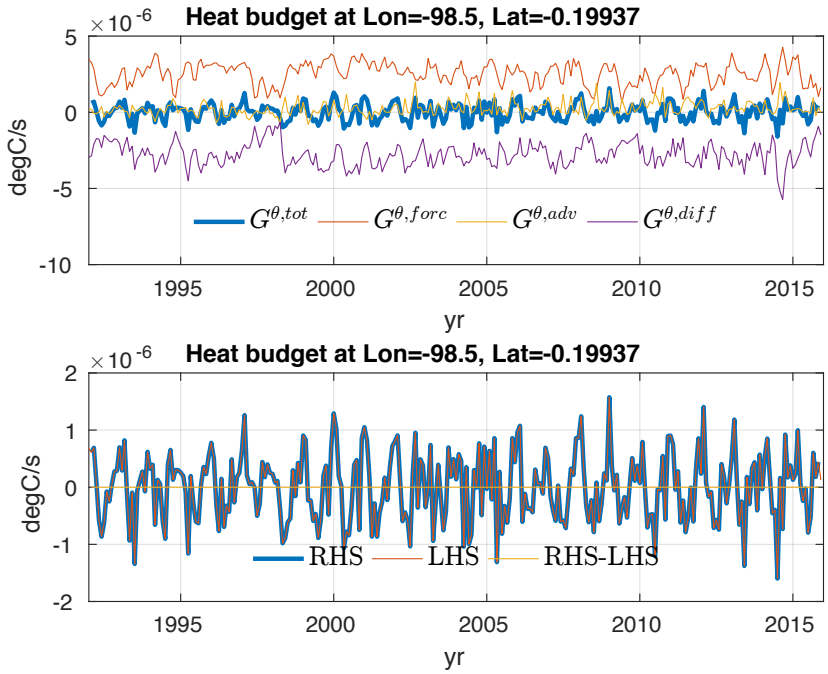

Figure 2: Heat budget for an arbitrary surface grid cell. Top panel shows the individual terms in the budget equation (4). Bottom panel shows the LHS, RHS, and difference between LHS and RHS terms in the budget. The good agreement between RHS and LHS (e.g., the ratio of the standard deviation of the residual to the standard deviation of the tendency here is $\mathcal{O}\left(5 \times 10^{-6}\right)$) demonstrates practical closure of the budget.

(and downloadable from the ECCO directory listing).

To demonstrate the relevance of this term in the global ocean heat budget, the horizontally averaged value of the geothermal heating is 0.095 W m^{-2}. This is not negligible relative to the average heating of the ocean in the ECCOv4 Release 3 solution over 1992–2015 (0.237 W m^{-2}).

To incorporate the geothermal contribution into the heat budget, one simply considers the ocean bottom grid cells, and normalizes the heat flux by reference density, specific heat capacity, and the vertical thickness of the bottom grid cell, as sketched in Algorithm (4).

4.3 Salt Conservation

The salt conservation equation in z^* is (see equation 5 in Forget et al. 2015),

$$\underbrace{\frac{\partial\left(s^{*} S\right)}{\partial t}}_{G^{S,tot}}=\underbrace{-\nabla_{z^{*}}\left(s^{*} S \mathbf{v}_{res}\right)-\frac{\partial\left(S w_{res}\right)}{\partial z^{*}}}_{G^{S,adv}}+\underbrace{s^{*} \mathcal{F}_{S}}_{G^{S,forc}}+\underbrace{s^{*} D_{S}}_{G^{S,diff}}, \tag{8}$$

Algorithm 4 : Including geothermal flux in the heat budget. See Algorithm (1) caption for description of subscript indices, grid parameters, and other symbols.

1. $Q^{geo}_{i,j}$ = `geothermalFlux.bin` ▷ 2-D time-invariant geothermal flux
   for $t = t_1, t_2, \ldots t_{T-1}, t_T$ **do** ▷ Loop over T time steps (months) t
      for $i = i_1, i_2, \ldots i_{I-1}, i_I$ **do** ▷ Loop over I longitude cells i
         for $j = j_1, j_2, \ldots j_{J-1}, j_J$ **do** ▷ Loop over J latitude cells j
            for $k = k_1, k_2, \ldots k_{K-1}, k_K$ **do** ▷ Loop over K vertical cells k
               if $k = k^{bot}_{i,j}$ **then** ▷ Do iff k is bottom cell at horizontal position (i, j)
                  $G^{\theta,forc}_{i,j,k} = Q^{geo}_{i,j} / (\rho_0 c_p h_{i,j,k} \Delta z_k)$
               end if
            end for
         end for
      end for
   end for

where S is salinity, and, in analogy with the heat budget equation (4), $\mathcal{F}_S$ and D_S are surface forcing and diffusive mixing of salt.

Given the time stepping, and again similar to the case of temperature, the discretized version of equation (8) relating the updated and current states $(\eta^{n+1}, \mathbf{v}^{n+1}, w^{n+1}, S^{n+3/2})$ and $(\eta^n, \mathbf{v}^n, w^n, S^{n+1/2})$ is (see equation B6 in Forget et al. 2015),

$$\frac{s^{*n+1} S^{n+3/2} - s^{*n} S^{n+1/2}}{\Delta t} = \mathcal{A}\left(S, \mathbf{u}^{n+1} + \mathbf{u}^{n+1}_b\right) + s^{*n}\left(\mathcal{F}^{n+1}_S + D^{n+1/2}_{\sigma,S} + D^{n+3/2}_{\perp,S}\right). \quad (9)$$

Table 4 lists `MITgcm` diagnostics needed for evaluating monthly salt budgets with ECCOv4. Evaluation of the total tendency and transport convergences in the salt budget (8) and (9) is performed in exactly the same manner as with the temperature budget (4) and (5). Therefore, we do not provide a separate pseudocode algorithm, but rather refer the reader to Algorithm (2), with appropriate replacements made between model diagnostics in Table 3 and those in Table 4 (e.g., `SALT` snapshots in place of `THETA` snapshots, and advection and diffusion diagnostics with suffix `_SLT` instead of `_TH`).

The local forcing term $G^{S,forc}$ reflects surface salt exchanges. As shown in Table 4, there are two relevant model diagnostics here, namely the total salt exchange at the surface (`SFLUX`), which is nonzero only when sea ice melts or freezes, and the salt plume tendency (`oceSPtnd`),

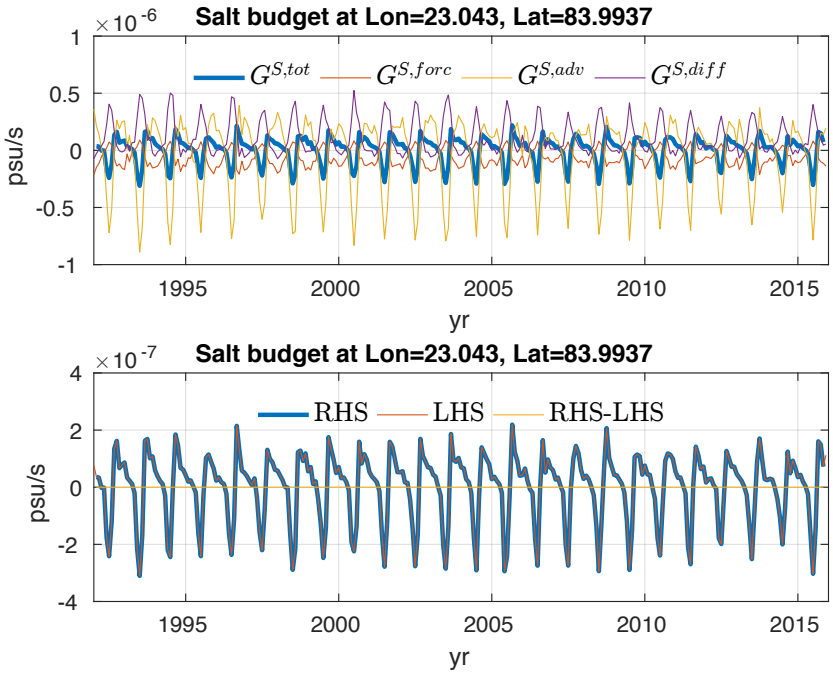

Figure 3: Salt budget for an arbitrary surface grid cell. Top panel shows the individual terms in the budget equation (8). Bottom panel shows the LHS, RHS, and difference between LHS and RHS terms in the budget. The good agreement between RHS and LHS (e.g., the ratio of the standard deviation of the residual to the standard deviation of the tendency here is $\mathcal{O}\,(2 \times 10^{-5})$) demonstrates practical closure of the budget.

which vertically redistributes surface salt input by sea ice formation following Duffy et al. (1999). A pseudocode sketch of an evaluation of the salt forcing term is given in Algorithm (5). (As before, we give specific Matlab code for evaluation of all terms in the salt budget in the Appendix.)

Figure 3 shows an example salt budget at an arbitrary grid cell using output in Table 4. Averaged over the global ocean surface in the first vertical layer ($k = 1$), the ratio of the standard deviation of the residual (LHS−RHS) to the tendency (LHS) in equation (8) using methods presented here is $\mathcal{O}\,(10^{-4})$.

An important point here is that, given the nonlinear free surface condition, budgets for *salt content* (an extensive quantity) are not the same as budgets for *salinity* (an intensive quantity). The attentive reader will have noticed that surface freshwater exchanges do not enter into salt budget equations, since such fluxes do not affect the overall salt content, but rather make it more or less concentrated. However, a budget for salinity can be derived based on the conservation equations for salt (8) and volume (2), and estimated using diagnostic model output. Such details are given in immediately below.

Diagnostic	Time	Description (Units)
`ETAN`	Snapshot	Surface height anomaly (m)
`SALT`	Snapshot	Salinity (psu)
`SFLUX`	Average	Total salt flux (g m^{-2} s^{-1})
`oceSPtnd`	Average	Salt tendency due to salt plume flux (g m^{-2} s^{-1})
`ADVr_SLT`	Average	Vertical advective flux of salinity (psu m^3 s^{-1})
`ADVx_SLT`	Average	Zonal advective flux of salinity (psu m^3 s^{-1})
`ADVy_SLT`	Average	Meridional advective flux of salinity (psu m^3 s^{-1})
`DFrI_SLT`	Average	Implicit vertical diffusive flux of salinity (psu m^3 s^{-1})
`DFrE_SLT`	Average	Explicit vertical diffusive flux of salinity (psu m^3 s^{-1})
`DFxE_SLT`	Average	Explicit zonal diffusive flux of salinity (psu m^3 s^{-1})
`DFyE_SLT`	Average	Explicit meridional diffusive flux of salinity (psu m^3 s^{-1})

Table 4: `MITgcm` diagnostics required to evaluate the grid cell salt budget.

Algorithm 5 : Evaluating the forcing term in the salt budget. See Algorithm (1) caption for description of subscript indices, grid parameters, and other symbols.

1. **for** $t = t_1, t_2, \ldots t_{T-1}, t_T$ **do** ▷ Loop over T time steps (months) t
   - $Q_{i,j} = \texttt{SFLUX}\,\{t\}$ ▷ 2-D **average** total surface salt flux over month t
   - $P_{i,j,k} = \texttt{oceSPtnd}\,\{t\}$ ▷ 3-D **average** salt plume tendency over month t
   - $\rho_0 = 1029$ ▷ Reference density (kg m^{-3})
   - **for** $i = i_1, i_2, \ldots i_{I-1}, i_I$ **do** ▷ Loop over I longitude cells i
     - **for** $j = j_1, j_2, \ldots j_{J-1}, j_J$ **do** ▷ Loop over J latitude cells j
       - **for** $k = k_1, k_2, \ldots k_{K-1}, k_K$ **do** ▷ Loop over K vertical cells k
         - $G^{S,forc}_{i,j,k} = 0$
         - **if** k=1 **then**
           - $G^{S,forc}_{i,j,k} = G^{S,forc}_{i,j,k} + Q_{i,j} / (\rho_0 h_{i,j,k} \Delta z_k)$
         - **end if**
         - $G^{S,forc}_{i,j,k} = G^{S,forc}_{i,j,k} + P_{i,j,k} / (\rho_0 h_{i,j,k} \Delta z_k)$
       - **end for**
     - **end for**
   - **end for**

5 Budgets Estimated for Intensive Quantities

Above, we walked through the model conservation equations for the extensive quantities volume, heat, and salt content, and demonstrated their offline evaluation. However, oftentimes interest is in an intensive quantity, such as salinity or density. As examples, below we derive a conservation equation for salinity in the rescaled height coordinate, and demonstrate how to estimate this budget offline using output from the ECCOv4 solution.

Yet, it is important to note at the outset that the budget derived and presented below does not correspond to a conservation equation diagnosed online by the model. As a result, there are some nonlinear product terms that appear in the equation that do not have corresponding available model diagnostics. Therefore, unlike with evaluation of the extensive property budgets above, small residual errors can be incurred in the offline estimation of the intensive property budget below.

5.1 Salinity Budget

Here we derive the salinity budget in the z^* coordinate, give a pseudocode sketch of evaluation of the budget using monthly model output. (Concrete Matlab code is given in the Appendix.)

We partition the LHS tendency in the salt conservation equation (8) using the product rule,

$$\frac{\partial\left(s^{*}S\right)}{\partial t}=s^{*}\frac{\partial S}{\partial t}+S\frac{\partial s^{*}}{\partial t}. \tag{10}$$

Substituting the sum of terms on the RHS of equation (10) for the LHS term in equation (8) and solving for $\partial S/\partial t$ gives an expression for the salinity tendency,

$$\frac{\partial S}{\partial t}=-\frac{1}{s^{*}}\left[S\frac{\partial s^{*}}{\partial t}+\nabla_{z^{*}}\left(s^{*}S\mathbf{v}_{res}\right)+\frac{\partial\left(Sw_{res}\right)}{\partial z^{*}}\right]+\mathcal{F}_{S}+D_{S}. \tag{11}$$

Noting that $\partial s^*/\partial t \equiv H^{-1}\partial\eta/\partial t$, we use the continuity equation (2) to cast equation (11) as,

$$\underbrace{\frac{\partial S}{\partial t}}_{G^{\dagger,tot}}=\underbrace{\frac{1}{s^{*}}\left[S\nabla_{z^{*}}\left(s^{*}\mathbf{v}\right)+S\frac{\partial w}{\partial z^{*}}-\nabla_{z^{*}}\left(s^{*}S\mathbf{v}_{res}\right)-\frac{\partial\left(Sw_{res}\right)}{\partial z^{*}}\right]}_{G^{\dagger,adv}}+\underbrace{\mathcal{F}_{S}-S\mathcal{F}}_{G^{\dagger,forc}}+\underbrace{D_{S}}_{G^{\dagger,diff}}. \tag{12}$$

Notice here that, in contrast to the salt content conservation equation (8), the surface forcing

term in the salinity equation (12) comprises both surface salt fluxes as well as surface freshwater fluxes (converted to appropriate units through multiplication by salinity).

Estimation of the salinity budget involves diagnostics given in Tables 2 and 4, with the addition of the monthly means of salinity (SALT) and surface height (ETAN). Budget evaluation roughly follows on from the basic steps outlined in Algorithms (1), (2), and (5), as sketched in Algorithm (6). Figure 4 shows an example salinity budget at an arbitrary sea surface grid cell. (Example Matlab code appears in the Appendix.) Averaged over the global ocean surface in the first vertical layer ($k = 1$), the ratio of the standard deviation of the residual (LHS−RHS) to the tendency (LHS) in equation (12) using methods presented here is $\mathcal{O}(10^{-3})$.

Algorithm 6 : Evaluating the salinity budget. See Algorithm (1) caption for description of subscript indices, grid parameters, and other symbols. This algorithm assumes that operations performed in Algorithm (1) for the volume budget and Algorithms (2) and (5) in the salt budget are still valid here (and are not repeated to save space).

1.
for $t = t_1, t_2, \ldots t_{T-1}, t_T$ **do** ▷ Loop over T time steps (months) t
  $S_{i,j,k} = \texttt{SALT}\{t\}$ ▷ 3-D **average** salinity over month t
  $N_{i,j} = \texttt{ETAN}\{t\}$ ▷ 2-D **average** surface height over month t
  for $i = i_1, i_2, \ldots i_{I-1}, i_I$ **do** ▷ Loop over I longitude cells i
    for $j = j_1, j_2, \ldots j_{J-1}, j_J$ **do** ▷ Loop over J latitude cells j
      $s^*_{i,j} = (1 + N_{i,j}/H_{i,j})$
      for $k = k_1, k_2, \ldots k_{K-1}, k_K$ **do** ▷ Loop over K vertical cells k
        $G^{\dagger,tot}_{i,j,k} = \left(S^{(f)}_{i,j,k} - S^{(0)}_{i,j,k}\right)/\Delta t$
        $G^{\dagger,adv}_{i,j,k} = \left(G^{S,adv}_{i,j,k} - S_{i,j,k} G^{\eta,conv}_{i,j,k}\right)/s^*_{i,j}$
        $G^{\dagger,diff}_{i,j,k} = \left(G^{S,forc}_{i,j,k} - S_{i,j,k} G^{\eta,forc}_{i,j,k}\right)/s^*_{i,j}$
        $G^{\dagger,forc}_{i,j,k} = G^{S,diff}_{i,j,k}/s^*_{i,j}$
      end for
    end for
  end for
end for

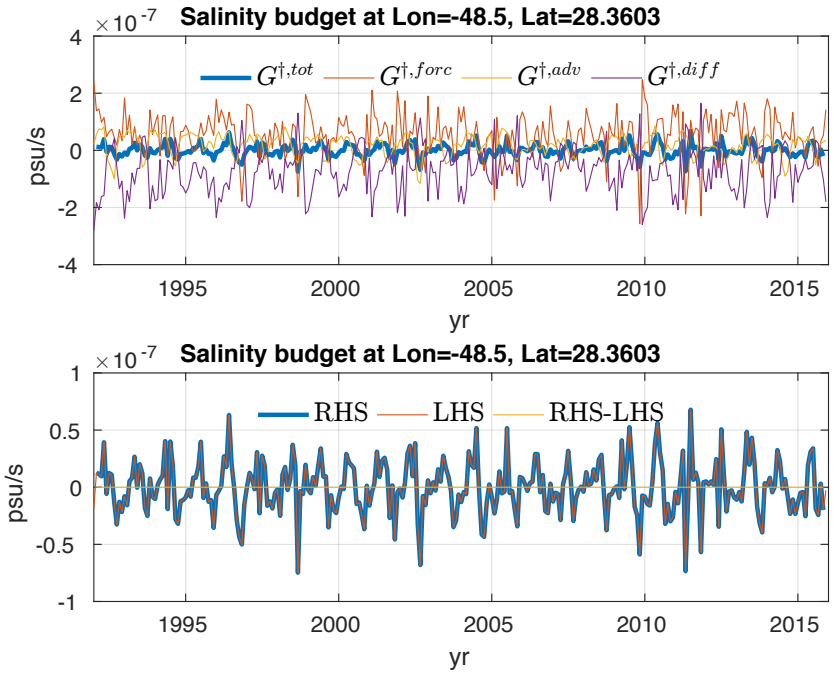

Figure 4: Salinity budget for an arbitrary surface grid cell. Top panel shows the individual terms in the budget equation (12). Bottom panel shows the LHS, RHS, and difference between LHS and RHS terms in the budget. The good agreement between RHS and LHS (e.g., the ratio of the standard deviation of the residual to the standard deviation of the tendency here is $\mathcal{O}(10^{-4})$) demonstrates practical closure of the budget.

A Example Matlab Code and the gcmfaces Framework

A.1 The gcmfaces Framework

The ECCOv4 estimates are provided on a native longitude-latitude-cap (LLC) grid topology. To allow for easy manipulation of the ECCOv4 output on the LLC grid and MITgcm output on all other grids, Gaël Forget at MIT has produced a suitable Matlab class and framework, called `gcmfaces`.

A current version of `gcmfaces` suitable for use with ECCOv4 can be found here. If they have not already done so, we recommend that the user download and read the `gcmfaces.pdf` document, which describes getting started with `gcmfaces`, including how to download, initialize, and update.

A.2 Example Matlab code

Here we provide example Matlab code for evaluating budgets, explaining our steps along the way, and relating to the conservation equations given above.

We assume the working directory is `/myDirectory/`. Within `/myDirectory/`, we assume the user has downloaded and initialized `gcmfaces` (as described in `gcmfaces.pdf`), and that the user has downloaded the relevant model diagnostics and stored them respectively in subdirectories `/nctiles_monthly/` and `/nctiles_monthly_snapshots/`. (See Tables 2, 3, and 4 above for the diagnostics.) Also, the user should have downloaded the ECCOv4 grid files, which are found here, and stored them in a subdirectory called `/nctiles_grid/` within `/myDirectory/gcmfaces/`. Further, we assume that the subdirectories `/budget_volume/`, `/budget_heat/`, `/budget_salt/`, and `/budget_salinity/` exist (and are empty) within `/myDirectory/`. Lastly, we assume that the reader has downloaded the `geothermalFlux.bin` from the ECCO directory listing, and placed this file in the subdirectory `/myDirectory/input_init/`.

1. The user begins by instantiating the `gcmfaces` framework and loading the ECCOv4 grid parameters, contained in the global `mygrid` structure (Box 1).

```
%%%%%%%%%%
% initialize workspace
clear all, close all, clc
cd /myDirectory/gcmfaces/
%%%%%%%%%%

%%%%%%%%%%
% instantiate gcmfaces and load grid
gcmfaces_global
global mygrid; mygrid=[];
grid_load;
%%%%%%%%%%
```

Box 1. Instantiation of `gcmfaces` and loading of model grid.

2. Next, for computing property tendencies from snapshot output, it can be helpful to define a number of parameters related to the time steps of the model output. ECCOv4 Release 3

is provided for the $nn = 288$ months over the period 1992-01-01 12:00:00 through 2015-12-31 12:00:00. The convention here is to define the time of a particular month of output (tt) as the end of the corresponding averaging period (in hr from the initial time). So, the first difference of the time (dt) is the number of hours in every month (Box 2). For example, for January 1994, the first difference of the time is $dt = 744$ hr.

```
%%%%%%%%%
% define monthly times over
% 1992-01-01 12:00:00 to 2015-12-31 12:00:00
nn=288;
tt=[1992*ones(nn,1) [2:(nn+1)]' [1*ones(nn-1,1); 0.5]];
tt=24*(datenum(tt)-datenum([1992 1 1 12 0 0]));
dt=diff([0 tt']);
ttUnits='hours since 1992-1-1 12:00:00';
secPerHour=3600;
%%%%%%%%%
```

Box 2. Definition of time parameters.

3. In addition to parameters related to time, it is also helpful for the user to define several quantities related to the grid's spatial geometry. In Box 3, after defining strings for the directories housing the nctiles output, we define several three-dimensional `gcmfaces`-class objects related to the depth (`dzMat`, `dzMatF`), surface area (`RACMat`), and volume (`VVV`) of each model spatial grid cell. These objects are used in subsequent computations of spatial integrals and averages. The `nLevels` variable, which is the number of vertical levels on the grid (here 50), is used in evaluations of the vertically penetrating shortwave radiation forcing in the heat budget. Additionally, we load in the geothermal flux forcing file and convert to a `gcmfaces` object.

```
%%%%%%%%%
% define directories and other useful fields
dir1='/myDirectory/nctiles_monthly/';
dir2='/myDirectory/nctiles_monthly_snapshots/';
dir3='/myDirectory/input_init/';
dzMatF=mk3D(mygrid.DRF, mygrid.hFacC);
dzMat=dzMatF.*mygrid.hFacC;
RACMat=mk3D(mygrid.RAC, mygrid.hFacC);
VVV=mygrid.mskC.*RACMat.*dzMat;
nLevels=numel(mygrid.RC);
%%%%%%%%%

%%%%%%%%%
% load 2d geothermal flux and make 3d

% load, reshape, and make gcmfaces
fid=fopen([dir3,'geothermalFlux.bin'],'r','b');
geoflx2d=fread(fid,'float32'); fclose(fid);
geoflx2d=reshape(geoflx2d,90,1170);
geoflx2d=convert2gcmfaces(geoflx2d);

% create 3d version
mskc=mygrid.mskC;
mskc(isnan(mskc))=0;
mskcp1=mskc;
mskcp1(:,:,nLevels+1)=0;
mskcp1(:,:,1)=[];
mskb=mskc-mskcp1;
geoflx3d=mk3D(geoflx2d,mskc).*mskb.*mygrid.mskC;
clear mskc mskcp1 mskb geoflx2d
%%%%%%%%%
```

Box 3. Definition of directories and space parameters.

4. ECCOv4 Release 3 file sizes can be large. To make it more feasible to load multiple state variables needed for budget calculations, we loop over the time steps, evaluating one month of output at a time (Box 4).

```
%%%%%%%%%
% loop through time steps
for ii=1:nn, disp(num2str(ii))
```

Box 4. Begin loop over nn time steps (i.e., months of output).

5. For each ii, we use the `read nctiles.m` function provided with `gcmfaces` to concatenate the nctiles and load into the Matlab workspace the monthly average diagnostic output needed for computing RHS transport-convergence and surface-forcing terms in the budgets of volume (Box 5), ...

```
%%%%%%%%%
% load 2-d monthly surface height and volume forcing
oceFWflx=read_nctiles([dir1,'oceFWflx'],'oceFWflx',ii);
ETAN=read_nctiles([dir1,'ETAN'],'ETAN',ii);

%%%%%%%%%
% load 3-d monthly volume-related fields
UVELMASS=read_nctiles([dir1,'UVELMASS'],'UVELMASS',ii);
VVELMASS=read_nctiles([dir1,'VVELMASS'],'VVELMASS',ii);
WVELMASS=read_nctiles([dir1,'WVELMASS'],'WVELMASS',ii);
```

Box 5. Loading monthly averaged variables for volume budget.

6. ... heat (Box 6), ...

```
%%%%%%%%%
% load 2-d monthly surface heat forcing
TFLUX=read_nctiles([dir1,'TFLUX'],'TFLUX',ii);
oceQsw=read_nctiles([dir1,'oceQsw'],'oceQsw',ii);

%%%%%%%%%
% load 3-d monthly heat-related fields
ADVr_TH=read_nctiles([dir1,'ADVr_TH'],'ADVr_TH',ii);
ADVx_TH=read_nctiles([dir1,'ADVx_TH'],'ADVx_TH',ii);
ADVy_TH=read_nctiles([dir1,'ADVy_TH'],'ADVy_TH',ii);
DFrI_TH=read_nctiles([dir1,'DFrI_TH'],'DFrI_TH',ii);
DFrE_TH=read_nctiles([dir1,'DFrE_TH'],'DFrE_TH',ii);
DFxE_TH=read_nctiles([dir1,'DFxE_TH'],'DFxE_TH',ii);
DFyE_TH=read_nctiles([dir1,'DFyE_TH'],'DFyE_TH',ii);
```

Box 6. Loading monthly averaged variables for heat budget.

7. ... and salt (Box 7).

```
%%%%%%%%%
% load 2-d monthly surface salt forcing
SFLUX=read_nctiles([dir1,'SFLUX'],'SFLUX',ii);

%%%%%%%%%
% load 3-d monthly salt-related fields
SALT=read_nctiles([dir1,'SALT'],'SALT',ii);
ADVr_SLT=read_nctiles([dir1,'ADVr_SLT'],'ADVr_SLT',ii);
ADVx_SLT=read_nctiles([dir1,'ADVx_SLT'],'ADVx_SLT',ii);
ADVy_SLT=read_nctiles([dir1,'ADVy_SLT'],'ADVy_SLT',ii);
DFrI_SLT=read_nctiles([dir1,'DFrI_SLT'],'DFrI_SLT',ii);
DFrE_SLT=read_nctiles([dir1,'DFrE_SLT'],'DFrE_SLT',ii);
DFxE_SLT=read_nctiles([dir1,'DFxE_SLT'],'DFxE_SLT',ii);
DFyE_SLT=read_nctiles([dir1,'DFyE_SLT'],'DFyE_SLT',ii);
oceSPtnd=read_nctiles([dir1,'oceSPtnd'],'oceSPtnd',ii);
```

Box 7. Loading monthly averaged variables for salt budget.

8. We also load the monthly snapshot diagnostic outputs for the start ($ii-1$) and end (ii)

of month ii needed for computing LHS tendency terms (Box 8). A note here is that, for the start of the first month ($ii = 1$), and the end of the last month ($ii = 288$), no snapshots are available. While this precludes calculation of the tendency terms for the first and last months based using snapshots, because the budgets close (for all practical purposes), as will be shown below, tendency terms for these months are in principle "recoverable" by summing up the various RHS convergence and forcing terms, as described previously.

```
%%%%%%%%%
% load snapshots for computing tendencies
if ii==1|ii==nn % no initial or final snapshots
 ETAN_SNAP=convert2gcmfaces(nan*ones(90,1170,2));
 THETA_SNAP=convert2gcmfaces(nan*ones(90,1170,nLevels,2));
 SALT_SNAP=convert2gcmfaces(nan*ones(90,1170,nLevels,2));
else
 THETA_SNAP=read_nctiles([dir2,'THETA'],'THETA',[(ii-1) ii]);
 SALT_SNAP=read_nctiles([dir2,'SALT'],'SALT',[(ii-1) ii]);
 ETAN_SNAP=read_nctiles([dir2,'ETAN'],'ETAN',[(ii-1) ii]);
end
```

Box 8. Loading monthly snapshots for volume, heat, and salt budgets.

9. With the model diagnostics loaded into the Matlab workspace, we assess terms in the volume budget equation (2). The tendency is computed by differencing `ETAN` snapshots corresponding to the start and end of the averaging period, dividing by the temporal "width" of the averaging period (dt), and scaling by a reference density, so units are kg m^{-2} (Box 9). The surface forcing term is simply the `oceFWflx` diagnostic. The horizontal transport convergence is computed by vertically integrating mass-weighted zonal and meridional velocity fields (`UVELMASS` and `VVELMASS`) and using the `gcmfaces` function `calc_UV_conv.m` to compute their convergence, whereas the vertical convergence is computed by taking the difference between `WVELMASS` values from one layer vertical interface to the next. The result is scaled by density and surface area. These tendency, forcing, and convergence fields are then saved out to file.

```
%%%%%%%%%
% volume budget
% useful quantities
rhoconst=1029;
heatcap=3994;
rcp=rhoconst*heatcap;

 % total tendency
        tendV=(1./mk3D(mygrid.Depth,mygrid.mskC)).*...
        mk3D((ETAN_SNAP(:,:,2)-ETAN_SNAP(:,:,1))/...
        (secPerHour*dt(ii)),mygrid.mskC);

% horizontal convergence
        hConvV=mygrid.mskC.*calc_UV_conv(UVELMASS,VVELMASS,...
        {'dh'})./(RACMat.*hFacC);

% vertical divergence
        vConvV=0*hConvV;
        for nz=1:nLevels, %disp(num2str(nz))
                nzp1=min([nz+1,nLevels]);
                vConvV(:,:,nz)=squeeze(WVELMASS(:,:,nzp1).*...
            double(nz~=nLevels)-WVELMASS(:,:,nz).*...
            double(nz~=1))./(dzMat(:,:,nz));
    end

% forcing
        forcV=mygrid.mskC.*mk3D(oceFWflx,mygrid.mskC)./...
        (dzMat*rhoconst);
    forcV(:,:,2:nLevels)=0*mygrid.mskC(:,:,2:nLevels);

% save output
DT=dt(ii);
save(['/myDirectory/budget_volume/',num2str(ii),'.mat'],...
     'tendV','*ConvV','forcV','DT')
%%%%%%%%%
```

Box 9. Evaluating terms in the vertically integrated volume budget.

10. Next we evaluate the heat budget (4) on the model grid cell. Evaluation of the heat budget is somewhat more complicated than the volume budget, and we breakdown the steps in detail. First, we evaluate the LHS tendency (Box 10). For the beginning and end of the averaging period, we use the ETAN and THETA snapshots to evaluate the $s^*\theta$ term (within the partial derivative) on the LHS of equation (4). Next, we then take the difference between their product at the start and end of the averaging period, and divide by the time difference between them, giving units of °C s^{-1}.

```
%%%%%%%%%
% grid cell heat budget
% total tendency
HC_snap=0*THETA_SNAP;
for jj=1:2
        HC_snap(:,:,:,jj)=(THETA_SNAP(:,:,:,jj).*...
        (1+mk3D(ETAN_SNAP(:,:,jj)./mygrid.Depth,dzMat)));
end
tendH=(HC_snap(:,:,:,2)-HC_snap(:,:,:,1))/...
     (secPerHour*dt(ii));
```

Box 10. Evaluating the tendency in the heat budget.

11. Second, we evaluate the ocean heat transport convergences on the RHS of equation (4), involving horizontal and vertical advective and diffusive fluxes (Box 11). We again use calc_UV_conv.m to compute the convergences of the explicit horizontal heat advection (ADVx_TH and ADVy_TH) and diffusion (DFxE_TH and DFyE_TH). Note that together ADVx_TH and ADVy_TH constitute the $s^*\theta\mathbf{v}_{res}$ term within the divergence operator on the RHS of equation (4) (Box 11). We loop through each level, computing the convergence in vertical heat advection (ADVr_TH) and diffusion (DFrE_TH and DFrI_TH). Note that ADVr_TH is the θw_{res} term on the RHS of (4) (Box 11).[1] All convergences are normalized by grid volume, VVV, giving units of °C s^{-1}.

[1]For interested readers, these calculations mirror online computations performed in the MITgcm subroutine gad_calc_rhs.F. Also, note that, for the vertical diffusion, there are two relevant model diagnostics, one computed *explicitly* (see gad_calc_rhs.F), the other *implicitly* (see MITgcm subroutine impldiff.F).

```
% horizontal convergence
adv_hConvH=calc_UV_conv(ADVx_TH,ADVy_TH)./VVV;
dif_hConvH=calc_UV_conv(DFxE_TH,DFyE_TH)./VVV;

% vertical convergence
adv_vConvH=0*tendH;
dif_vConvH=0*tendH;
for nz=1:nLevels, %disp(num2str(nz))
        nzp1=min([nz+1,nLevels]);
        adv_vConvH(:,:,nz)=squeeze(ADVr_TH(:,:,nzp1)*...
        double(nz~=nLevels)-ADVr_TH(:,:,nz));
        dif_vConvH(:,:,nz)=squeeze(DFrI_TH(:,:,nzp1)*...
        double(nz~=nLevels)-DFrI_TH(:,:,nz)+...
        DFrE_TH(:,:,nzp1)*double(nz~=nLevels)-...
        DFrE_TH(:,:,nz));
end
adv_vConvH=adv_vConvH./VVV;
dif_vConvH=dif_vConvH./VVV;
```

Box 11. Evaluating the transport convergences in the heat budget.

12. Third, and finally, we evaluate the local forcing term due to surface heat exchanges and geothermal fluxes. For the surface contribution, there are two relevant model diagnostics here, the total heat flux (`TFLUX`) and its shortwave component (`oceQsw`). Given the penetrating nature of the shortwave term, to properly evaluate the local forcing term in Matlab, `oceQsw` must be removed from `TFLUX` (which contains the net latent, sensible, longwave, and shortwave contributions) and redistributed vertically following (6) and (7).

In Box 12, we take the first steps, defining the relevant constants in equations (6) and (7). Note that the values of q_1 and q_2 are "zeroed out" below 200 m depth, as the shortwave radiation does not penetrate below this depth.

```
% surface heat flux
% note that shortwave penetrates the top 200 m
% constants
c_p=3994;
rho0c_p=rho0*c_p;
R=0.62;
zeta1=0.6;
zeta2=20;
q1=R*exp(1/zeta1*mygrid.RF(1:nLevels))+...
    (1-R)*exp(1/zeta2*mygrid.RF(1:nLevels));
q2=R*exp(1/zeta1*mygrid.RF(2:(nLevels+1)))+...
    (1-R)*exp(1/zeta2*mygrid.RF(2:(nLevels+1)));

% correction for the 200m cutoff
zCut=find(mygrid.RC<-200,1);
q1(zCut:nLevels)=0;
q2((zCut-1):nLevels)=0;
```

Box 12. Defining terms needed for evaluating surface heat forcing.

13. Having defined the necessary constants, we loop through each level, subtracting `oceQsw` from `TFLUX` at the surface and redistributing `oceQsw` vertically (Box 13). After the geothermal component at the seafloor is added in, the local forcing term is normalized by the grid cell vertical thickness and the product of density and heat capacity, giving units of °C s^{-1}, and the output saved to file (Box 13).

```
% compute vertically penetrating flux
forcH=0*tendH;
msk=mygrid.mskC; msk(isnan(msk))=0;
for nz=1:nLevels, %disp(num2str(nz))
        if nz==1
                forcH(:,:,nz)=TFLUX(:,:,1) -...
            (1-(q1(nz)-q2(nz)))*oceQsw;
        else
                nzp1=min([nz+1,nLevels]);
                forcH(:,:,nz)=forcH(:,:,nz)+...
            ((mygrid.mskC(:,:,nz)==1).*q1(nz) -...
            (mygrid.mskC(:,:,nzp1)==1).*...
            q2(nz)).*oceQsw;
        end
end
% add geothermal
forcH=forcH+geoflx3d;
forcH=mygrid.mskC.*forcH./(rho0c_p*dzMat);

% save output
save(['/myDirectory/budget_heat/',num2str(ii),'.mat'],...
     'tendH','*ConvH','forcH','DT')
%%%%%%%%%
```

Box 13. Evaluating the local forcing term in the heat budget.

14. Next, we evaluate the salt budget equation (8). We again walk through the evaluation of the tendency, convergence, and forcing terms step by step. These steps to the salt budget are very similar to the steps to the heat budget. First, we assess the LHS tendency (Box 14). We use `ETAN` and `SALT` snapshots from the start and end of the averaging period to evaluate the s^*S term on the LHS of (8). We take the difference between their product at the start and end of the averaging period, and divide by time difference, yielding units of psu s^{-1}.

```
%%%%%%%%%
% grid cell salt budget
% total tendency
HC_snap=0*SALT_SNAP;
for jj=1:2
        HC_snap(:,:,:,jj)=(SALT_SNAP(:,:,:,jj).*...
        (1+mk3D(ETAN_SNAP(:,:,jj)./mygrid.Depth,...
        dzMat)));
end
tendS=(HC_snap(:,:,:,2)-HC_snap(:,:,:,1))/...
    (secPerHour*dt(ii));
```

Box 14. Evaluating the tendency in the salt budget.

15. Second, we evaluate ocean salt transport convergences on the RHS of (8), involving horizontal and vertical advective and diffusive fluxes (Box 15). We use `calc_UV_conv.m` to compute the convergences of explicit horizontal heat advection (`ADVx_SLT` and `ADVy_SLT`) and diffusion (`DFxE_SLT` and `DFyE_SLT`). As before, together `ADVx_SLT` and `ADVy_SLT` constitute the $s^* S \mathbf{v}_{res}$ term on the RHS of (8) (Box 15). We loop through each level, computing the convergence in vertical salt advection (`ADVr_SLT`) and diffusion (`DFrE_SLT` and `DFrI_SLT`). `ADVr_SLT` is $S w_{res}$ on the RHS of (8). All convergences are normalized by grid volume, `VVV`, giving units of psu s^{-1}.

```
% horizontal divergences
adv_hConvS=calc_UV_conv(ADVx_SLT,ADVy_SLT)./VVV;
dif_hConvS=calc_UV_conv(DFxE_SLT,DFyE_SLT)./VVV;

% vertical divergences
adv_vConvS=0*tendS;
dif_vConvS=0*tendS;
for nz=1:nLevels, %disp(num2str(nz))
        nzp1=min([nz+1,nLevels]);
        adv_vConvS(:,:,nz)=squeeze(ADVr_SLT(:,:,nzp1)*...
        double(nz~=nLevels)-ADVr_SLT(:,:,nz));
        dif_vConvS(:,:,nz)=squeeze(DFrI_SLT(:,:,nzp1)*...
        double(nz~=nLevels)-DFrI_SLT(:,:,nz)+...
        DFrE_SLT(:,:,nzp1)*double(nz~=nLevels)-...
        DFrE_SLT(:,:,nz));
end
adv_vConvS=adv_vConvS./VVV;
dif_vConvS=dif_vConvS./VVV;
```

Box 15. Evaluating the transport convergences in the salt budget.

13. Third, and finally, we evaluate the local forcing term due to surface salt exchanges (Box 16). There are two relevant model diagnostics here, the total salt flux (`SFLUX`), which is nonzero only when sea ice melts or freezes, and the salt plume tendency (`oceSPtnd`), which vertically redistributes salt rejected by sea-ice formation, following Duffy et al. (1999) and Nguyen et al. (1999). The local forcing term is normalized by the grid cell vertical thickness and density, giving units of psu s^{-1}, and the output is saved (Box 16). An example of the budget from these calculations at an arbitrary grid cell is shown in Figure 3.

```
% surface salt flux
% note that salt (plume) flux penetrates vertically
forcS=0*tendS;
for nz=1:nLevels
        if nz==1
                forcS(:,:,nz)=SFLUX/rho0;
        end
        forcS(:,:,nz)=forcS(:,:,nz)+...
        oceSPtnd(:,:,nz)/rho0;
end
forcS=forcS./(dzMat);

% save output
save(['/myDirectory/budget_salt/',num2str(ii),'.mat'],...
    'tendS','*ConvS','forcS','DT')
%%%%%%%%%
```

Box 16. Evaluating the local forcing term in the salt budget.

17. Based on the above volume and salt budgets, the salinity budget can be evaluated as per equation (12), as shown in Box 17.

```
% salinity budget based on salt and volume budgets
% scale factor
rstarfac=(mygrid.Depth+ETAN)./mygrid.Depth;

% tendency
tendSln=(SALT_SNAP(:,:,:,2)-SALT_SNAP(:,:,:,1))/...
    (secPerHour*(dt(ii)));

% advection
adv_vConvSln=(-SALT.*vConvV+adv_vConvS)./rstarfac;
adv_hConvSln=(-SALT.*hConvV+adv_hConvS)./rstarfac;

% diffusion
dif_vConvSln=(dif_vConvS)./rstarfac;
dif_hConvSln=(dif_hConvS)./rstarfac;

% forcing
forcSln=(-SALT.*forcV+forcS)./rstarfac;

% save output
 save(['/myDirectory/budget_salinity/',num2str(ii),...
     '.mat'],'tendSln','*ConvSln','forcSln','DT')
```

Box 17. Evaluating the salinity budget.

18. Finally, the end of the loop is reached, and some variables cleared.

```
 % clear tendencies, convergences, and forcing
 clear tend* *Conv* forc*
end
```

Box 18. end loop and clear tendencies, convergences, and forcing.

Acknowledgements

Ichiro Fukumori, Jean-Michel Campin, Ian Fenty, Gaël Forget, and Rui Ponte gave helpful comments on initial drafts of this memo. Martha Buckley, Nadya Vinogradova, and Katy Quinn made valuable contributions to the Matlab code provided in the Appendix.

References

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

- Duffy, P. B., M. Eby, and A. J. Weaver, 1999: Effects of sinking of salt rejected during formation of sea ice on results of an ocean-atmosphere-sea ice climate model. *Geophys. Res. Lett.*, *26*, 12, 1739–1742.

- 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. *Geosci. Model Dev.*, *8*, 3071-3104.

- Gaspar, P., Y. Grégoris, and J.-M. Lefevre, 1990: A Simple Eddy Kinetic Energy Model for Simulations of the Oceanic Vertical Mixing: Tests at Station Papa and Long-Term Upper Ocean Study Site. *J. Geophys. Res.*, *95*, C9, 16179–16193.

- Griffies, S. M., and R. J. Greatbatch, 2012: Physical processes that impact the evolution of global mean sea level in ocean climate models. *Ocean Modell.*, *51*, 37–73.

- Marshall, J., A. Adcroft, C. Hill, L. Perelman, and C. Heisey, 1997: A finite-volume, incompressible Navier Stokes model for studies of the ocean on parallel computers. *J. Geophys. Res.*, *102*, C3, 5753–5766.

- Paulson, C. A., and J. J. Simpson, 1977: Irradiance Measurements in the Upper Ocean. *J. Phys. Oceanogr.*, *7*, 952–956.

- Piecuch, C. G., P. Heimbach, R. M. Ponte, and G. Forget, 2015: Sensitivity of contemporary sea level trends in a global ocean state estimate to effects of geothermal fluxes. *Ocean Modell.*, *96*, 214–220.

- Thompson, P. R., C. G. Piecuch, M. A. Merrifield, J. P. McCreary, and E. Firing, 2016: Forcing of recent decadal variability in the Equatorial and North Indian Ocean. *J. Geophys. Res.-Oceans*, *121*, doi:10.1002/ 2016JC012132.