Part VII

Miscellaneous

Chapter 23

Basics of the finite difference method

This chapter describes the basics of finite difference methods for solving differential equations. The general principles of the finite differencing methods are introduced using the diffusion equation as an example in Section 23.1. Sections 23.2 and 23.3 describe applying finite difference methods of basic time and space derivatives in differential equations. A more sophisticated time-integration method will be explained later in 23.6. Considerations in finite-difference methods for advection-diffusion equations are discussed in Section 23.4. An implicit method for solving the diffusion equation is described in Section 23.5. In 23.6, we mainly discuss the errors of various time-integration methods for the wave equation. A comprehensive description of these differencing methods in geophysical fluid dynamics is given in detail in Durran (2010).

23.1 Diffusion equation

As an example, consider an initial-boundary value problem expressed by a one-dimensional diffusion equation (heat conductive equation),

$$\frac{\partial T}{\partial t} = \kappa \frac{\partial^2 T}{\partial x^2}. \tag{23.1}$$

Given $T(x,0) = f(x)$ as the initial distribution and $T(0,t) = T(L,t) = 0$ as the boundary condition, the analytical solution is

$$T(x,t) = \sum_{m=0}^{\infty} f_m e^{-\kappa k_m^2 t} \sin(k_m x), \tag{23.2}$$

where

$$f_m = \frac{2}{L}\int_0^L f(x)\sin(k_m x)dx, \quad k_m = \frac{m\pi}{L}. \tag{23.3}$$

Next, consider the finite difference method to get the solution numerically. In the finite difference method, grids are set with a finite increment in space and time, and each term in the equation is evaluated at each grid using $T_j^n = T(x_j, t_n)$. For example,

$$\frac{T_j^{n+1} - T_j^n}{\Delta t} = \kappa \frac{T_{j+1}^n - 2T_j^n + T_{j-1}^n}{\Delta x^2}, \tag{23.4}$$

where $\Delta t = t_{n+1} - t_n$ and $\Delta x = x_{j+1} - x_j$.

Distribution at the new time level T^{n+1} can then be calculated if T^n is known. This finite difference equation is identical to the original differential equation (23.1) in the limit $\Delta t \to 0, \Delta x \to 0$ (**consistency**).

If the initial distribution is assumed to be $f(x) = T_0 \sin k_1 x$, the solution of the finite difference equation (23.4) for $t = t_n$ is

$$T_j^n = \lambda^n T_0 \sin k_1 x_j, \tag{23.5}$$

where

$$\lambda = 1 - \frac{2\kappa\Delta t}{\Delta x^2}(1 - \cos k_1 \Delta x). \tag{23.6}$$

In order to suppress oscillation and divergence of the solution (**stability**), $0 < \lambda < 1$ is necessary and Δx and Δt must be set to satisfy this condition. This solution is identical to the analytical solution in the limit of $\Delta t \to 0, \Delta x \to 0$ (**convergence**).

To summarize, the finite difference method that satisfies consistency, stability, and convergence is the necessary condition for an accurate solution.

23.2 Finite difference expressions for time derivatives

The following are the four basic finite difference expressions. These had been used in the MRI.COM before ver. 4.0.

$$\text{forward}: \frac{T^{n+1}-T^{n}}{\Delta t} = F(T^{n}) \tag{23.7}$$

$$\text{backward}: \frac{T^{n+1}-T^{n}}{\Delta t} = F(T^{n+1}) \tag{23.8}$$

$$\text{Matsuno}: \frac{T^{*n+1}-T^{n}}{\Delta t} = F(T^{n}), \quad \frac{T^{n+1}-T^{n}}{\Delta t} = F(T^{*n+1}) \tag{23.9}$$

$$\text{leap-frog}: \frac{T^{n+1}-T^{n-1}}{2\Delta t} = F(T^{n}). \tag{23.10}$$

The scheme used in the previous section is the forward scheme. The forward, backward, and Matsuno schemes use the values at two time levels and are accurate to $O(\Delta t)$, while the leap-frog scheme uses three time levels and is accurate to $O(\Delta t)^2$. Basically, the leap-frog scheme is employed in MRI.COM because of its higher order accuracy.

However, the leap-frog scheme cannot be applied to the diffusion equation. A solution by the finite difference method using the leap-frog scheme is given by

$$T_j^n = (T_a\lambda_a^n + T_b\lambda_b^n)\sin k_1 x_j, \tag{23.11}$$

where

$$\lambda_a = -\frac{-\alpha + \sqrt{\alpha^2+4}}{2}, \quad \lambda_b = -\frac{-\alpha - \sqrt{\alpha^2+4}}{2} \quad \left(\alpha \equiv \frac{4\kappa\Delta t}{\Delta x^2}(1-\cos k_1\Delta x)\right). \tag{23.12}$$

Because $\lambda_b < -1$ for arbitrary values of α, the divergent mode with oscillation is always included (**computational mode**). In order to avoid this computational mode, the forward scheme is employed for diffusion and viscosity terms in MRI.COM.

When the diffusion and viscosity coefficients are very large as in the surface mixed layer, the time step has to be unusually small for the stability of the forward scheme according to (23.6). In such a case, the backward scheme is used for vertical diffusion and viscosity (implicit method; see Section 23.5). Though the time integration at each point can proceed without referring to the result of other points by the forward, leap-frog, and Matsuno schemes, it must be done by solving combined linear equations in the backward scheme (see Section 23.5).

The Matsuno scheme is useful for suppressing the computational mode in the leap-frog scheme. By defaults, the Matsuno scheme is used once per twelve steps of the leap-frog scheme in MRI.COM. In the old version, this interval can be changed at run time using a namelist parameter (`nstep_matsuno_interval`) of namelist `nml_time_step` (Table 25.6). It should be noted that the Matsuno scheme needs twice as many numerical operations as the forward and leap-frog schemes.

The simplest forward method is used in components such as the sea ice model, mixed layer model, and ecosystem model. The 3rd-order leap-frog Adams-Moulton scheme (LFAM3) used in the current version of the MRI.COM dynamical frame is discussed in the next section, mainly considering the errors of various time integration methods for the wave equation.

23.3 Finite difference expression for space derivatives

Let us consider a one-dimensional advection equation,

$$\frac{\partial T}{\partial t} = -u\frac{\partial T}{\partial x}, \tag{23.13}$$

where u is a constant velocity. The solution is

$$T(x,t) = T(x-ut,0). \tag{23.14}$$

Using the leap-frog scheme for time differencing, the finite difference equation can be written as follows:

$$\frac{T_j^{n+1}-T_j^{n-1}}{2\Delta t} = -u\frac{T^n_{j+\frac{1}{2}}-T^n_{j-\frac{1}{2}}}{\Delta x}, \tag{23.15}$$

where $T^n_{j-\frac{1}{2}}$ and $T^n_{j+\frac{1}{2}}$ are the values at the left and right faces of the grid cell for x_j, i.e., values at $x_{j-\frac{1}{2}}$ and $x_{j+\frac{1}{2}}$. The point $x_{j-\frac{1}{2}}$ is defined as the central point between x_j and x_{j-1}. Because the transport of T at the boundary that enters a grid cell is identical to that leaving the adjacent grid cell, the total T in the whole system is conserved in this finite difference equation.

There are several methods to decide $T^n_{j-\frac{1}{2}}$ using a value at a single or multiple grid points. The following are two simple and frequently used formulations,

$$\text{upstream finite difference}: T^n_{j-\frac{1}{2}} = T^n_{j-1}\ (u > 0), \quad T^n_{j-\frac{1}{2}} = T^n_j\ (u < 0), \tag{23.16}$$

$$\text{central finite difference}: T^n_{j-\frac{1}{2}} = \frac{T^n_{j-1} + T^n_j}{2}. \tag{23.17}$$

The former is accurate to $O(\Delta x)$, and the latter is accurate to $O(\Delta x^2)$.

In central finite differencing, the expression for (23.15) is

$$\frac{T^{n+1}_j - T^{n-1}_j}{2\Delta t} = -u\frac{T^n_{j+1} - T^n_{j-1}}{2\Delta x}. \tag{23.18}$$

Assuming the solution to be $T(x,t) = \tau(t)e^{-ikx}$,

$$\tau^{n+1} = \tau^{n-1} + 2i\alpha\tau^n, \quad \text{where } \alpha \equiv \frac{u\Delta t}{\Delta x}\sin k\Delta x. \tag{23.19}$$

It is stable (neutral) if $|\alpha| \le 1$. To be stable for any wave number,

$$\left|\frac{u\Delta t}{\Delta x}\right| \le 1 \tag{23.20}$$

must be satisfied (**CFL condition**). However, if $\tau^n = \tau^0 e^{-in\Delta\theta}$,

$$\Delta\theta = -\sin^{-1}[\mu\sin k\Delta x](\text{where } \mu \equiv \frac{u\Delta t}{\Delta x}). \tag{23.21}$$

Expanding the r.h.s. by a Taylor expansion we obtain

$$\begin{aligned}\Delta\theta &\simeq -\mu\sin k\Delta x - \frac{1}{6}(\mu\sin k\Delta x)^3\\ &\simeq -\mu k\Delta x + \frac{\mu(k\Delta x)^3}{6} - \frac{\mu^3(k\Delta x)^3}{6}\\ &= -\mu k\Delta x\left\{1 - \frac{(k\Delta x)^2}{6}(1-\mu^2)\right\}.\end{aligned} \tag{23.22}$$

This means that the phase of the solution from this finite difference scheme is delayed relative to that of analytical solution, depending on its wavenumber (**numerical dispersion**). Therefore, a distribution with maxima and minima that do not exist in the initial distribution arises. However, this method is popularly used since the kinetic energy is conserved by employing the central difference in the advection term in the equation of motion. Moreover, the "Arakawa method," which can nearly conserve the enstrophy (squared vorticity) for horizontally non-divergent flows, is adopted in MRI.COM by using the central difference. This topic is treated in Chapter 8.

Using the upstream finite difference, the finite difference equation (23.15) is

$$\frac{T^{n+1}_j - T^{n-1}_j}{2\Delta t} = -u\frac{T^n_j - T^n_{j-1}}{\Delta x}. \tag{23.23}$$

Expanding the r.h.s. by a Taylor expansion we obtain

$$-u\frac{\partial T}{\partial x} + \frac{u\Delta x}{2}\frac{\partial^2 T}{\partial x^2} + O(\Delta x^2). \tag{23.24}$$

The second term has the diffusion (heat conductive) form (which disappears in the central finite differencing). Actually, the initial distribution diffuses when the advection equation is solved by the upstream finite difference (**numerical diffusion**).

The third order schemes (QUICK, QUICKEST, and UTOPIA) can be used in MRI.COM to suppress the numerical dispersion and diffusion somewhat in the advection calculation for tracers, but not completely. The grid boundary value is set in QUICK as

$$T^n_{j-\frac{1}{2}} = \frac{-T^n_{j-2} + 6T^n_{j-1} + 3T^n_j}{8}(u > 0), \quad T^n_{j-\frac{1}{2}} = \frac{3T^n_{j-1} + 6T^n_j - T^n_{j+1}}{8}(u < 0). \tag{23.25}$$

The QUICKEST method uses the time averaged value at the grid boundary as the tracer value to be transported, considering the change of the value there by advection during one time step. UTOPIA is a multi-dimensional extension of QUICKEST. The details of these schemes are described in Chapter 10.

23.4 Finite differencing of advection-diffusion equation

According to the above restriction, when the advection-diffusion equation (Eqs. (10.1) and (11.1)) is expressed in finite difference form using the leap-frog scheme, it is necessary to use present (previous) time level for advection (diffusion) term. The following finite difference equation is then employed:

$$\frac{T^{n+1}-T^{n-1}}{2\Delta t} = -\mathcal{A}(T^{n}) + \mathcal{D}(T^{n-1}), \tag{23.26}$$

where $\mathcal{A}$ and $\mathcal{D}$ are advection and diffusion operators, respectively.

When the vertical diffusion term is very large, $\mathcal{D}(T^{n+1})$ is used instead of $\mathcal{D}(T^{n-1})$. This formula is an implicit scheme and is described in the next section.

23.5 Implicit method for vertical diffusion equation

Turbulent mixing is parameterized by using high vertical diffusivity and viscosity determined by boundary layer models, which was treated in Chapter 15. The time step must be set very small to keep the calculation stable when viscosity and diffusivity are very high, since the time tendency becomes very large due to the rapid mixing. To avoid this problem, the implicit method uses the advanced (mixed) state for evaluating viscosity and diffusivity, unlike the normal explicit method where previous or present values are used.

Expressing the present time step as n and the time steps before and after as $n \pm 1$, the finite-difference method is applied to the advection-diffusion equation using the leap-frog scheme. The diffusion term is written separately using $(n-1)$ step for the horizontal direction and $(n+1)$ step for the vertical direction,

$$\frac{(T^{n+1}-T^{n-1})}{2\Delta t} = -\mathcal{A}(T^{n}) + \mathcal{D}_H(T^{n-1}) + \mathcal{D}_V(T^{n+1}). \tag{23.27}$$

Putting all the terms involving T^{n+1} on the l.h.s.,

$$T^{n+1} - 2\Delta t\mathcal{D}_V(T^{n+1}) = T^{n-1} + 2\Delta t(-\mathcal{A}(T^{n}) + \mathcal{D}_H(T^{n-1})) \tag{23.28}$$

is obtained, which is an algebraic equation for T^{n+1}. The finite difference form is rewritten specifically as

$$\begin{aligned} T_k^{n+1} - 2\Delta t\frac{1}{\Delta z_k}\left(\kappa_{k-\frac{1}{2}}(T_{k-1}^{n+1}-T_k^{n+1})/\Delta z_{k-\frac{1}{2}} - \kappa_{k+\frac{1}{2}}(T_k^{n+1}-T_{k+1}^{n+1})/\Delta z_{k+\frac{1}{2}}\right) \\ = T_k^{n-1} + 2\Delta t(-\mathcal{A}(T_k^{n}) + \mathcal{D}_H(T_k^{n-1})). \end{aligned} \tag{23.29}$$

By putting

$$a = \frac{2\Delta t\kappa_{k-\frac{1}{2}}}{\Delta z_k \Delta z_{k-\frac{1}{2}}}, \quad b = 1+a+c, \quad c = \frac{2\Delta t\kappa_{k+\frac{1}{2}}}{\Delta z_k \Delta z_{k+\frac{1}{2}}}, \tag{23.30}$$

we get,

$$-aT_{k-1}^{n+1} + bT_k^{n+1} - cT_{k+1}^{n+1} = T_k^{n-1} + 2\Delta t(-\mathcal{A}(T_k^{n}) + \mathcal{D}_H(T_k^{n-1})). \tag{23.31}$$

Setting $\mathcal{F}_k \equiv -\mathcal{A}(T_k^{n}) + \mathcal{D}_H(T_k^{n-1})$, this is expressed in the matrix form as

$$\begin{pmatrix} b & -c & & & & & \\ -a & b & -c & & & & \\ & -a & b & -c & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & -a & b & -c & \\ & & & & -a & b & -c \\ & & & & & -a & b \end{pmatrix} \begin{pmatrix} T_1^{n+1} \\ T_2^{n+1} \\ T_3^{n+1} \\ \vdots \\ T_{KM-2}^{n+1} \\ T_{KM-1}^{n+1} \\ T_{KM}^{n+1} \end{pmatrix} = \begin{pmatrix} T_1^{n-1} + 2\Delta t\mathcal{F}_1 \\ T_2^{n-1} + 2\Delta t\mathcal{F}_2 \\ T_3^{n-1} + 2\Delta t\mathcal{F}_3 \\ \vdots \\ T_{KM-2}^{n-1} + 2\Delta t\mathcal{F}_{KM-2} \\ T_{KM-1}^{n-1} + 2\Delta t\mathcal{F}_{KM-1} \\ T_{KM}^{n-1} + 2\Delta t\mathcal{F}_{KM} \end{pmatrix}. \tag{23.32}$$

The l.h.s. has the form of the tri-diagonal matrix.

23.5.1 A solution of tri-diagonal matrix

In general, simultaneous linear equations for n variables with tri-diagonal matrix coefficients

$$\begin{pmatrix} B_1 & C_1 & & & \\ A_2 & B_2 & C_2 & & \\ & \ddots & \ddots & \ddots & \\ & & A_{n-1} & B_{n-1} & C_{n-1} \\ & & & A_n & B_n \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \\ \vdots \\ X_{n-1} \\ X_n \end{pmatrix} = \begin{pmatrix} D_1 \\ D_2 \\ \vdots \\ D_{n-1} \\ D_n \end{pmatrix} \tag{23.33}$$

are solved using the Thomas method, which is modified from LU decomposition,

$$P_1 = C_1/B_1 \tag{23.34}$$

$$Q_1 = D_1/B_1 \tag{23.35}$$

$$P_k = \frac{C_k}{B_k - A_k P_{k-1}} \qquad (2 \le k \le n-1) \tag{23.36}$$

$$Q_k = \frac{D_k - A_k Q_{k-1}}{B_k - A_k P_{k-1}} \qquad (2 \le k \le n) \tag{23.37}$$

$$X_n = Q_n \tag{23.38}$$

$$X_k = Q_k - P_k X_{k+1} \qquad (1 \le k \le n-1). \tag{23.39}$$

23.6 Evaluation of time-integration methods for wave equations

In the ocean models, the CFL condition is mainly determined by the speed of baroclinic gravity waves. In evaluating time integration in such wave propagation, it is not sufficient to evaluate only one variable as we have done in the previous sections, but it is necessary to consider the interaction of two variables. Here, we evaluate time integration methods for the simplest one-dimensional wave equation.

Consider the following one-dimensional wave equation,

$$\frac{\partial p}{\partial t} = -c\frac{\partial u}{\partial x}, \tag{23.40}$$

$$\frac{\partial u}{\partial t} = -c\frac{\partial p}{\partial x}. \tag{23.41}$$

We will discuss the accuracy of the time integration method based on Shchepetkin and McWilliams (2005) (hereafter, SM2005). For the parts not explained in SM2005, see Durran (2010), Section 2.2 for more details.

Applying the Fourier transformation with respect to space by placing $p(t,x) = \hat{p}_k e^{ikx}, u(t,x) = \hat{u}_k u^{ikx}$ and $\omega_k = kc$ yields,

$$\frac{\partial \hat{p}_k}{\partial t} = -i\omega_k \hat{u}_k, \tag{23.42}$$

$$\frac{\partial \hat{u}_k}{\partial t} = -i\omega_k \hat{p}_k. \tag{23.43}$$

For simplicity, we omit $\hat{}$ and the subscript k, and denote the variable ζ^n at the time $t = n\Delta t$ of the variable ζ. To further simplify the equation, we introduce the dimensionless quantity $\alpha = \omega_k \Delta t = ck\Delta t$ (Durran (2010) uses $\omega\Delta t$ throughout his text though it is a bit redundant). Since the maximum k (k_{max}) that can be resolved in the model is $1/\Delta x$, the maximum α (α_{max}) is $c\Delta t/\Delta x$, i.e. Courant number.

In studying the stability and accuracy of the time integration scheme for the wave equation, we assume that a variable ζ is multiplied by the complex number λ between n and $n+1$ steps. (Durran (2010) uses A instead of λ).

$$\zeta^{n+1} = \lambda\zeta^n. \tag{23.44}$$

During the time step, the amplification rate is $|\lambda| = \left(\Re(\lambda)^2 + \Im(\lambda)^2\right)^{1/2}$, and phase change is $\theta = \arctan(\Im(\lambda)/\Re(\lambda))$. ($\Re(), \Im()$ are functions to extract real and complex numbers, respectively). The relative phase error is estimated as $R = \theta/\alpha$ using the positive phase α. If $R > 1$, the scheme is accelerating; if $R < 1$, the scheme is decelerating. If the wave is completely represented, then $\lambda = e^{i\alpha}$. In SM2005, to visually represent the error, $\lambda(\alpha)$ is drawn on the complex plane by varying α and compared with the complete solution $e^{i\alpha}$ (i.e. the unit circle on the complex plane).

23.6.1 LFAM3 method

Here, we consider solving the wave equation with the third-order leap-frog Adams-Moulton method (LFAM3) as the time-integration method used in the current version of MRI.COM. In the LFAM3 method, we first obtain a tentative value of $p^{n+1,*}, u^{n+1,*}$ in the *predictor step*,

$$p^{n+1,*} = p^{n-1} - 2i\alpha u^n, \tag{23.45}$$

$$u^{n+1,*} = u^{n-1} - 2i\alpha p^n. \tag{23.46}$$

In the *corrector step*, we conduct the final time integration,

$$p^{n+1} = p^n - i\alpha \left\{ \frac{5}{12} u^{n+1,*} + \frac{2}{3} u^n - \frac{1}{12} u^{n-1} \right\}, \tag{23.47}$$

$$u^{n+1} = u^n - i\alpha \left\{ \frac{5}{12} p^{n+1,*} + \frac{2}{3} p^n - \frac{1}{12} p^{n-1} \right\}. \tag{23.48}$$

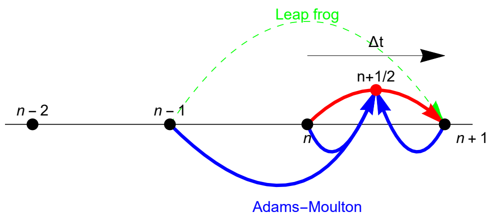

Figure23.1 Schematics of the LFAM3 time integration. The 3rd-order Adams-Moulton method uses values at $n-1$, n, and, $n+1$ steps to find the value at $n+\frac{1}{2}$ step. The LFAM3 method used the leap-frog method to create the value at $n+1$ step. This is why it is named LFAM3. For reference the third-order Adams-Bashforth methods uses values at $n-2$, $n-1$, and n steps to find the value at $n+\frac{1}{2}$.

The LFAM3 used for OGCM is as follows, which corresponds to SM2005 equations (2.38)-(2.41) (or SM2009 equations (4.1)-(4.2)). This is composed of the predictor step,

$$p^{n+1/2} = \left(\frac{1}{2} - 2\gamma\right) p^{n-1} + \left(\frac{1}{2} + 2\gamma\right) p^n - i\alpha(1-2\gamma)u^n \tag{23.49}$$

$$u^{n+1/2} = \left(\frac{1}{2} - 2\gamma\right) u^{n-1} + \left(\frac{1}{2} + 2\gamma\right) u^n - i\alpha(1-2\gamma)\left[p^n + \beta\frac{2p^{n+1/2} - 3p^n + p^{n-1}}{1-2\gamma}\right] \tag{23.50}$$

and corrector step,

$$p^{n+1} = p^n - i\alpha u^{n+1/2} \tag{23.51}$$

$$u^{n+1} = u^n - i\alpha \left\{ (1-\epsilon)p^{n+1/2} + \epsilon\left[\left(\frac{1}{2} - \gamma\right) p^{n+1} + \left(\frac{1}{2} + 2\gamma\right) p^n - \gamma p^{n-1}\right]\right\}. \tag{23.52}$$

In the case of $\gamma = 1/12, \beta = 0, \epsilon = 0$, (23.45–23.48) and (23.49–23.52) are mathematically equivalent. The procedure for (23.49–23.52) is described in detail in Section 3.

Now let us return to (23.45–23.48) to evaluate their errors. (23.45–23.48) is combined into two equations.

$$p^{n+1} = \left(1 - \frac{5}{6}\alpha^2\right) p^n - i\alpha \left\{ \frac{2}{3} u^n + \frac{1}{3} u^{n-1} \right\}, \tag{23.53}$$

$$u^{n+1} = \left(1 - \frac{5}{6}\alpha^2\right) u^n - i\alpha \left\{ \frac{2}{3} p^n + \frac{1}{3} p^{n-1} \right\}. \tag{23.54}$$

The above equations can be written in matrix form.

$$\begin{pmatrix} p^{n+1} \\ u^{n+1} \end{pmatrix} = \begin{pmatrix} 1-\frac{5}{6}\alpha^2 & -i\frac{2}{3}\alpha \\ -i\frac{2}{3}\alpha & 1-\frac{5}{6}\alpha^2 \end{pmatrix} \begin{pmatrix} p^n \\ u^n \end{pmatrix} + \begin{pmatrix} 0 & -i\frac{1}{3}\alpha \\ -i\frac{1}{3}\alpha & 0 \end{pmatrix} \begin{pmatrix} p^{n-1} \\ u^{n-1} \end{pmatrix} \tag{23.55}$$

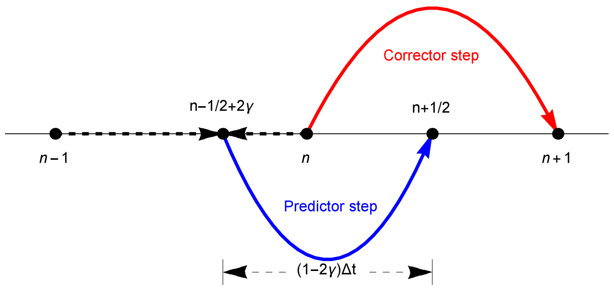

Figure23.2 Schematics of LFAM3 method used in the model. Mathematically this is equivalent to Fig. 23.1. The starting point of the predictor step is $n-1/2+2\gamma$ step, which is a linear combination of $n-1$ and n steps. The values at $n+\frac{1}{2}$ step is obtained by integrating over the interval $(1-2\gamma)\Delta t$. In the corrector step, the values at $n+1$ step is obtained from n by integrating over the domain n-$n+1$. The values at $n+\frac{1}{2}$ step are discarded after the corrector step.

The characteristic equation of this system is obtained by substituting $p^{n+1}=\lambda p^n=\lambda^2 p^{n-1}$, $u^{n+1}=\lambda u^n=\lambda^2 u^{n-1}$.

$$\lambda^2\begin{pmatrix}p^{n-1}\\u^{n-1}\end{pmatrix}=\lambda\begin{pmatrix}1-\frac{5}{6}\alpha^2 & -i\frac{2}{3}\alpha\\-i\frac{2}{3}\alpha & 1-\frac{5}{6}\alpha^2\end{pmatrix}\begin{pmatrix}p^{n-1}\\u^{n-1}\end{pmatrix}+\begin{pmatrix}0 & -i\frac{1}{3}\alpha\\-i\frac{1}{3}\alpha & 0\end{pmatrix}\begin{pmatrix}p^{n-1}\\u^{n-1}\end{pmatrix}\tag{23.56}$$

Rearranging it yields the following equation.

$$\begin{pmatrix}\lambda^2-\lambda(1-\frac{5}{6}\alpha^2) & i\alpha(\frac{2}{3}\lambda+\frac{1}{3})\\ i\alpha(\frac{2}{3}\lambda+\frac{1}{3}) & \lambda^2-\lambda(1+\frac{5}{6}\alpha^2)\end{pmatrix}\begin{pmatrix}p^{n-1}\\u^{n-1}\end{pmatrix}=\begin{pmatrix}0\\0\end{pmatrix}\tag{23.57}$$

From the above determinant = 0, we obtain the following solution

$$\lambda^2-\left(1-\frac{5}{6}\alpha^2\right)\lambda=\pm i\alpha\left(\frac{2}{3}\lambda+\frac{1}{3}\right)\tag{23.58}$$

The plus and minus in the equation correspond to the propagation of the waves in both directions, and placing $\alpha=-\alpha$ does not change the equation. Also, since the form is the same for p and u, the properties of the wave equation in LFAM are the same for that in LFAM3 for one variable.

Adopting the plus of equation (23.58), we obtain the following equation,

$$\lambda^{\pm}=\frac{1}{12}\left(g_1\pm\sqrt{(-g_1)^2+48i\alpha}\right)\tag{23.59}$$

$$g_1=6+4i\alpha-5\alpha^2\tag{23.60}$$

Physical mode and Computational mode are λ^+ and λ^- for the range $0<\alpha<1.55$, respectively. In the range $1.55<\alpha<1.59$ they are λ^- and λ^+ respectively.

Figure 23.3 shows the $\lambda(\alpha)$ of LFAM3 drawn in the complex plane by varying α and compared to the exact solution $e^{i\alpha}$ (i.e., the unit circle in the complex plane). Stability is determined by the amplitude of the physical mode exceeding 1, $\alpha=1.587$ at this time. Though computational mode decreases faster than physical mode, there is a drawback that the amplitudes of the computational mode and the physical mode approach each other when α exceeds 1.

23.6.2 Generalized Leap-Frog Adams-Mouton method (G-LFAM3)

SM2005 has shown that the accuracy and stability of the time integration can be increased for the wave equation by making minor modifications over the original LFAM3. The characteristic equation of Generalized LFAM3 derived by SM2005 is as follows (SM2005 Equation 2.31),

$$\begin{pmatrix}p^{n+1}\\u^{n+1}\end{pmatrix}=\begin{pmatrix}A & -iB\\-iC & D\end{pmatrix}\begin{pmatrix}p^{n}\\u^{n}\end{pmatrix}+\begin{pmatrix}E & -iF\\-iG & H\end{pmatrix}\begin{pmatrix}p^{n-1}\\u^{n-1}\end{pmatrix}\tag{23.61}$$

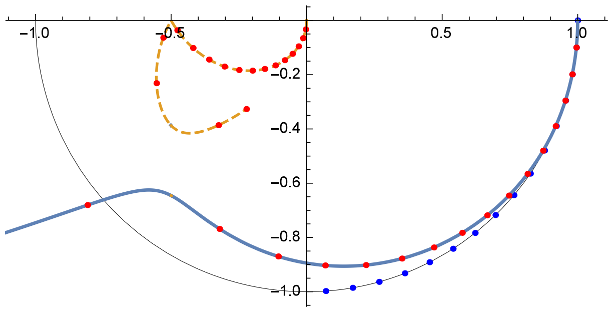

Figure23.3 Characteristic solution of LFAM3 ($\beta = \epsilon = 0, \gamma = 1/12$) on the complex plane. The solid line from (1,0) is the physical mode, and the dashed line from (0,0) is the computational mode. Dots are marked every time α changes by 0.1. The point where the amplitude in physical mode exceeds 1 is at $\alpha = 1.587$.

Here, A–H are

$$A = 1 - 2\alpha^2\left(\frac{1}{2} - \gamma\right)(1 - 2\beta), \qquad B = \alpha\left\{\frac{1}{2} + 2\gamma - 4\alpha^2\left(\frac{1}{2} - \gamma\right)\beta\right\}, \tag{23.62}$$

$$C = \alpha\left\{\frac{1}{2} + 2\gamma + \epsilon\left(\frac{1}{2} - \gamma\right)\left[1 - 2\alpha^2\left(\frac{1}{2} - \gamma\right)(1 - 2\beta)\right]\right\}, \tag{23.63}$$

$$D = 1 - 2\alpha^2\left(\frac{1}{2} - \gamma\right)\left\{1 - \epsilon\left[\frac{3}{4} - \gamma + 2\alpha^2\left(\frac{1}{2} - \gamma\right)\beta\right]\right\}, \tag{23.64}$$

$$E = -4\alpha^2\left(\frac{1}{2} - \gamma\right)\beta, \qquad F = \alpha\left(\frac{1}{2} - 2\gamma\right), \tag{23.65}$$

$$H = -\alpha^2\left(\frac{1}{2} - \gamma\right)\left(\frac{1}{2} - 2\gamma\right)\epsilon, \qquad G = \alpha\left\{\frac{1}{2} - 2\gamma - \epsilon\left(\frac{1}{2} - \gamma\right)\left[1 + 4\alpha^2\left(\frac{1}{2} - \gamma\right)\beta\right]\right\} \tag{23.66}$$

SM2005 discusses how to move γ, ϵ, β to reduce error and increase stability. According to this, the case $\gamma = 1/12, \epsilon = 11/20, \beta = 17/120$ has the smallest error: its accuracy for the wave equation is $O(\alpha^4)$. In this case the solution is very close to the ideal solution as shown in Fig. 23.4. Stability is determined by the amplitude of computational mode exceeding 1, where $\alpha = 1.851640$. This is the default setting of MRI.COM. The most stable combination is $\gamma = 1/12, \beta = 0.126, \epsilon = 0.83$ (Fig. 23.5). In this case the place where the amplitude of computational mode exceeds 1 is $\alpha = 1.958537$, but its accuracy for the wave equation is $O(\alpha^3)$.

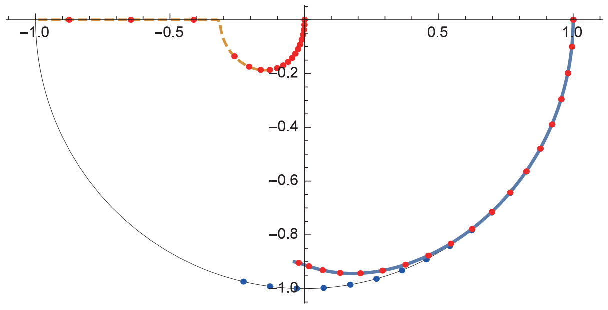

Figure23.4 Characteristic solutions of G-LFAM3 in the complex plane for the most accurate case $\gamma = 1/12, \beta = 17/120, \epsilon = 11/20$. Its accuracy for the wave equation is $O(\alpha^4)$. The solid line from (1,0) is the physical mode, and the dashed line from (0,0) is the computational mode. The point is marked every time α changes by 0.1, and the place where the amplitude of the computational mode exceeds 1 is $\alpha = 1.851640$.

23.6.3 Evaluation of errors in leap-frog method for wave equation

Here, we estimate the error in solving the wave equation with leap frog method previously used in MRI.COM.

$$p^{n+1} = -2i\alpha u^n + p^{n-1} \tag{23.67}$$

$$u^{n+1} = -2i\alpha p^n + u^{n-1} \tag{23.68}$$

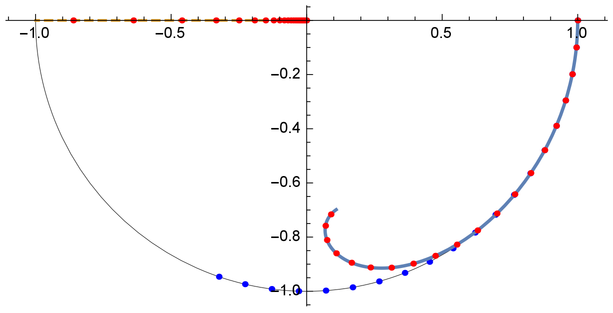

Figure23.5 Characteristic solutions of G-LFAM3 in the complex plane for the most stable case $\gamma = 1/12, \beta = 0.126, \epsilon = 0.83$. Its accuracy for the wave equation is $O(\alpha^3)$. The solid line following from (1,0) is the physical mode, and the dashed line following from (0,0) is the computational mode. Dots are marked every time α changes by 0.1. The point where the amplitude of the computational mode exceeds 1 is $\alpha = 1.958537$.

The following equation is obtained when written using matrices.

$$\begin{pmatrix} p^{n+1} \\ u^{n+1} \end{pmatrix} = \begin{pmatrix} 0 & -2i\alpha \\ -2i\alpha & 0 \end{pmatrix} \begin{pmatrix} p^{n} \\ u^{n} \end{pmatrix} + \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} p^{n-1} \\ u^{n-1} \end{pmatrix} \tag{23.69}$$

The characteristic equations of this system are obtained by substituting $p^{n+1} = \lambda p^n = \lambda^2 p^{n-1}$, $u^{n+1} = \lambda u^n = \lambda^2 u^{n-1}$.

$$\lambda^2 \begin{pmatrix} p^{n-1} \\ u^{n-1} \end{pmatrix} = \lambda \begin{pmatrix} 0 & -2i\alpha \\ -2i\alpha & 0 \end{pmatrix} \begin{pmatrix} p^{n-1} \\ u^{n-1} \end{pmatrix} + \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} p^{n-1} \\ u^{n-1} \end{pmatrix} \tag{23.70}$$

This is rewritten as

$$\begin{pmatrix} \lambda^2 - 1 & 2i\alpha\lambda \\ 2i\alpha\lambda & \lambda^2 - 1 \end{pmatrix} \begin{pmatrix} p^{n-1} \\ u^{n-1} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}. \tag{23.71}$$

From the determinant = 0, we obtain the following equation,

$$(\lambda^2 - 1)^2 + 4\alpha^2\lambda^2 = 0. \tag{23.72}$$

$$(\lambda^2 - 1) = \pm 2i\alpha\lambda \tag{23.73}$$

From the negative sign (the result is the same for the positive sign), we obtain the characteristic equation,

$$\lambda^2 + 2i\alpha\lambda - 1 = 0. \tag{23.74}$$

The characteristic equation above is exactly the same as the characteristic equation derived from the following one-variable leap-frog equation.

$$p^{n+1} = p^{n-1} - 2i\alpha p^n \tag{23.75}$$

$$\lambda^2 + 2i\alpha\lambda - 1 = 0 \tag{23.76}$$

The quadratic formula yields

$$\lambda = -i\alpha \pm \sqrt{1 - \alpha^2} \tag{23.77}$$

However, the above expression is correct only for $0 < \alpha < 1$. Of these, $-i\alpha + \sqrt{1-\alpha^2}$ is called the physical mode and $-i\alpha - \sqrt{1-\alpha^2}$ is called the computational mode.

The amplification factor $|\lambda|$ is always 1 in the range $0 < \alpha < 1$. The relative phase shift is written as *

$$R = \frac{1}{\alpha} \arctan\left(\frac{\alpha}{\sqrt{1-\alpha^2}}\right) \simeq 1 + \frac{\alpha^2}{6} + \frac{3\alpha^4}{40} + O(\alpha^5) \tag{23.78}$$

From Fig. 23.6, we can see that the phase is accelerated in the leap-frog time integration scheme. Since the amplitude of not only the physical mode but also the computational mode is 1, it is necessary to introduce some mechanism to attenuate

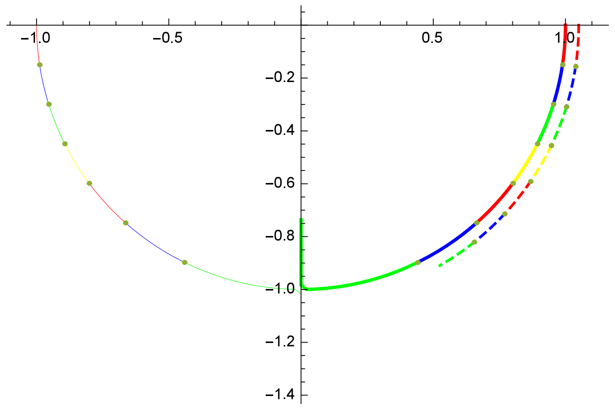

Figure23.6 Characteristic solutions in the complex plane of LF. The line starting from (1,0) is the physical mode, and the line starting from (-1,0) is the computational mode. The outer dotted line is the exact solution ($e^{i\alpha}$). The dotted line outside is the exact solution ($e^{i\alpha}$). Originally, the two lines overlap, but the exact solution is shifted outward to make the difference in phase easier to understand. The color is changed every time α changes $\pi/18$.

the computational mode. In the old version of MRI.COM, this was handled by inserting the Matsuno scheme about once every 10 times.

23.6.4 Leap frog with Shuman's averaging (LF-SA)

The method discussed in this subsection is not employed by MRI.COM, but is used by NEMO and others and is explained for comparison. This method adopts Shuman's averaging (LF-SA), which uses a weighted average of $n+1$, $n-1$, and n steps to calculate the pressure gradient. In this case, it is known that the time step can be set to be twice as long as that of a normal leap-frog method for wave equation. Here, we confirm this by actually solving the wave equations with the Shuman's averaging method (LF-SA), which are written as

$$p^{n+1} = p^{n-1} - 2i\alpha u^n, \tag{23.79}$$

$$u^{n+1} = u^{n-1} - 2i\alpha \left(\frac{p^{n-1} + 2p^n + p^{n+1}}{4}\right). \tag{23.80}$$

The characteristic equation in the matrix form is

$$\lambda^2 \begin{pmatrix} p^{n-1} \\ u^{n-1} \end{pmatrix} = \begin{pmatrix} 1 & -2i\alpha\lambda \\ -2i\alpha\left(\frac{1+2\lambda+\lambda^2}{4}\right) & 1 \end{pmatrix} \begin{pmatrix} p^{n-1} \\ u^{n-1} \end{pmatrix} \tag{23.81}$$

From the above determinant = 0, the eigenvalues are solutions of the following characteristic equations.

$$(\lambda^2 - 1)^2 + \alpha^2\lambda(\lambda+1)^2 = 0 \tag{23.82}$$

This equation has multiple solutions as $\lambda = -1$, which is the computational mode. The physical mode is

$$\lambda = \frac{2 - \alpha^2 \pm i\alpha\sqrt{4-\alpha^2}}{2}. \tag{23.83}$$

The amplification factor $|\lambda|$ is always 1 in the range $0 < \alpha < 2$, and its relative phase error is

$$R = \frac{1}{\alpha} \arctan\left(\frac{\alpha\sqrt{4-\alpha^2}}{2-\alpha^2}\right) \simeq 1 + \frac{\alpha^2}{24} + \frac{3\alpha^4}{640} + O(\alpha^5). \tag{23.84}$$

This is practically the same as replacing $\alpha' = \alpha/2$ in the LF method.

* The derivation of the last Taylor expansion is complicated. Here, it was derived using mathematica. The relative phase accuracy is $O(\alpha)$ and the absolute phase accuracy is $O(\alpha^2)$.

23.6.5 Time-staggered method

The method discussed in this subsection is not employed by MRI.COM, but is used by MITgcm, MOM, and others and is explained for comparison. The properties of this solution are the same as the physical mode of LF-SA according to Lemariè et al. (2015). Here, we confirm this by actually solving the wave equations with the time-staggered method, which are written as

$$p^{n+\frac{1}{2}} = p^{n-\frac{1}{2}} - i\alpha u^n, \tag{23.85}$$

$$u^{n+1} = u^{n-1} - i\alpha p^{n+\frac{1}{2}}. \tag{23.86}$$

In matrix form, it is rewritten as

$$\lambda \begin{pmatrix} p^{n-1/2} \\ u^n \end{pmatrix} = \begin{pmatrix} 1 & -i\alpha \\ -i\alpha\lambda & 1 \end{pmatrix} \begin{pmatrix} p^{n-\frac{1}{2}} \\ u^n \end{pmatrix} \tag{23.87}$$

Its characteristic equation is

$$(\lambda - 1)^2 + \alpha^2 \lambda = 0. \tag{23.88}$$

This solution has no computational mode. Its physical mode is

$$\lambda = \frac{2 - \alpha^2 \pm i\alpha\sqrt{4 - a^2}}{2}. \tag{23.89}$$

This solution is indeed identical to the physical mode of LF-SA shown in the previous subsection.

23.6.6 The 3rd-order Adams-Bashforth method (AB3)

The third-order Adams-Bashforth method (AB3), which may be used in the time extrapolation of the external force for the barotropic component of the momentum equation, is also evaluated here with a single variable.

$$q^{n+\frac{1}{2}} = \frac{23}{12} q^n - \frac{4}{3} q^{n-1} + \frac{5}{12} q^{n-2}, \tag{23.90}$$

$$q^{n+1} = q^n - i\alpha q^{n+\frac{1}{2}}. \tag{23.91}$$

The characteristic equation is

$$\lambda^3 = \left(1 - i\alpha\frac{23}{12}\right)\lambda^2 + i\alpha\frac{4}{3}\lambda - i\alpha\frac{5}{12}. \tag{23.92}$$

Analytical solutions exist, but they are very long and have been omitted. As shown in Fig. 23.7, there are physical and computational modes. The instability occurs when the amplitude of the computational mode exceeds 1 at $\alpha = 0.72$. It has $O(\alpha^3)$ accuracy.

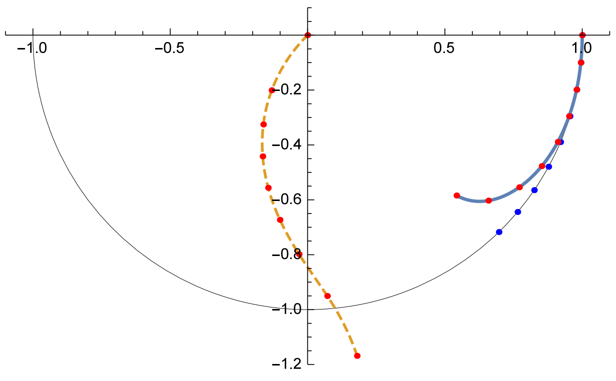

Figure23.7 The characteristic solution of the AB3 method in the complex plane. The solid line starting from (1,0) is the physical mode, and the dashed line starting from (0,0) is the computational mode. The points are marked every time α changes by 0.1. The place where the amplitude of the computational mode exceeds 1 is $\alpha = 0.72$.

23.6.7 Quasi-second-order Adams-Bashforth method (AB2ϵ)

The quasi-second-order Adams-Bashforth method (AB2ϵ), which may be used in the time extrapolation of external forces for the forward pressure equation, is also evaluated for a single variable. Quasi-second order means that the original AB2 is second-order accurate but unstable, so by introducing ϵ, stability is obtained instead of giving up perfect $O(\Delta t^2)$ accuracy.

$$q^{n+\frac{1}{2}} = \left(\frac{3}{2}+\epsilon\right)q^n - \left(\frac{1}{2}+\epsilon\right)q^{n-1} \tag{23.93}$$

$$q^{n+1} = q^n - i\alpha q^{n+\frac{1}{2}} \tag{23.94}$$

The characteristic equation and its solution are

$$\lambda^2 = \left\{1 - i\alpha\left(\frac{3}{2}+\epsilon\right)\right\}\lambda + i\alpha\left(\frac{1}{2}+\epsilon\right), \tag{23.95}$$

$$\lambda^{\pm} = \frac{1}{4}\left(g \pm \sqrt{g^2 + 8i\alpha(1+2\epsilon))}\right) \tag{23.96}$$

$$g = -i\alpha(3+2\epsilon)+2 \tag{23.97}$$

The λ^+, λ^- are the physical mode and computational mode, respectively. As shown in Fig. 23.8, the physical mode is distributed almost on the unit circle and slightly outside the unit circle around $\alpha = 0.50$. The amplitude of the computational mode is kept small enough. When ϵ=0, it is outside the unit circle at the starting point of (1,0).

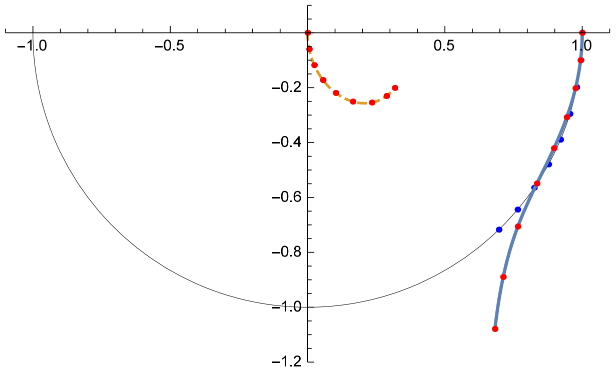

Figure23.8 The characteristic solution of the AB2ϵ method in the complex plane. The solid line starting from (1,0) is the physical mode, and the dashed line starting from (0,0) is the computational mode. The points are marked every time α changes by 0.1. The place where the amplitude of the computational mode exceeds 1 is α = 0.50254.

23.6.8 Matsuno scheme

Here, we evaluate the Matsuno scheme, which was previously used with the LF method on MRI.COM.

$$q^{n+1*} = q^n - i\alpha q^n \tag{23.98}$$

$$q^{n+1} = q^n - i\alpha q^{n+1*} = \left(1 - i\alpha - \alpha^2\right)q^n \tag{23.99}$$

Its characteristic solution is

$$\lambda = 1 - i\alpha - \alpha^2 \tag{23.100}$$

Though it is stable for $0 < \alpha < 1$, it has large amplitude and phase errors (Fig. 23.9).

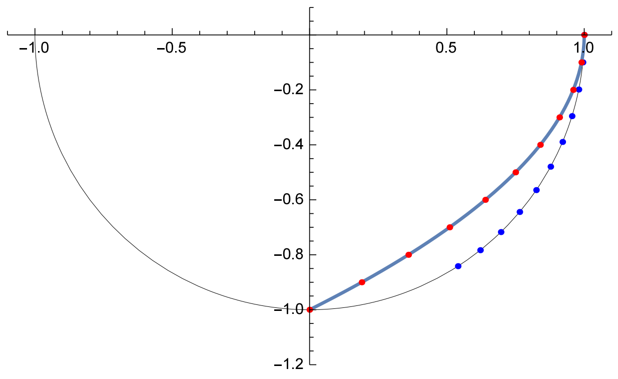

Figure23.9 The characteristic solution of the Matsuno method in the complex plane. The points are marked every time α changes by 0.1. The place where its amplitude exceeds 1 is $\alpha = 1$.

23.7 Evaluation of the time-integration method for advection-diffusion equation

Until now, we have concentrated on stability analysis for waves (internal gravity waves) according to SM2005, so we have evaluated the time-integration method by considering only ω_i in the following equation,

$$\frac{\partial \hat{q}_k}{\partial t} = -\omega \hat{q}_k, \tag{23.101}$$

$$\omega = \omega_r + i\omega_i, \tag{23.102}$$

and moving $\alpha \equiv \omega_i \Delta t$ in the range of real numbers. Then the method has been evaluated by comparing the behavior of the eigenvalues (λ) of the characteristic equation with the analytical solution ($e^{i\alpha}$).

This is equivalent to placing $M = 0$ in the following one-dimensional advection-diffusion equation.

$$\frac{\partial q}{\partial t} + c\frac{\partial q}{\partial x} = M\frac{\partial^2 q}{\partial x^2} \tag{23.103}$$

On the other hand, Lemariè et al. (2015) considers that ω_r can be used to evaluate the difference between upstream differential (numerical diffusion) and central differential advection in stability analysis. To do so, he investigated the behavior of the eigenvalues (λ) of the characteristic equation by moving ω on the complex plane. Following this method, in this section we examine stability analysis for various time-difference schemes, i.e., the range of amplitudes $|\lambda| \le 1$ of the characteristic equations. This is equivalent to the analysis called Absolute Stability (A-Stability) in Durran (2010). Here we perform A-Stability for the LFAM3 method in the one-variable case and show that the imaginary part of this solution is equivalent to the analysis of SM2005. The time integral of LFAM3 can be expressed for the Fourier series $\hat{q}$ using $\mu \equiv \omega \Delta t$ as follows

$$\hat{q}^{n+1,*} = \hat{q}^{n-1} + 2\mu \hat{q}^n, \qquad (LF) \tag{23.104}$$

$$\hat{q}^{n+1/2} = \frac{5}{12}\hat{q}^{n+1,*} + \frac{2}{3}\hat{q}^n - \frac{1}{12}\hat{q}^{n-1} \qquad \text{(AM3)} \tag{23.105}$$

$$\hat{q}^{n+1} = \hat{q}^n + \mu \hat{q}^{n+1/2} \qquad \text{(corrector)}, \tag{23.106}$$

$$= \left(1 + \frac{2\mu}{3} + \frac{5\mu^2}{6}\right)\hat{q}^n + \frac{\mu}{3}\hat{q}^{n-1}. \tag{23.107}$$

The characteristic equation for (23.107) is

$$\lambda^2 = \left(1 + \frac{2\mu}{3} + \frac{5\mu^2}{6}\right)\lambda + \frac{\mu}{3}. \tag{23.108}$$

Its solution is

$$\lambda^{\pm} = \frac{1}{12}\left(g_1 \pm \sqrt{(-g_1)^2 + 48\mu}\right) \tag{23.109}$$

$$g_1 = 6 + 4\mu + 5\mu^2 \tag{23.110}$$

The region of A-Stability is obtained by finding the region of $|\lambda| \leq 1$ on the complex plane of μ, which corresponds to the interior of the contour in Fig. 23.10. On the other hand, if we set $\mu_r = 0$ in λ and vary λ on the complex plane with $\alpha \equiv \mu_i$ as a parameter, we obtain Fig. 23.3. In fact, substituting $\mu = -i\alpha$ in equation (23.108), we obtain equation (23.60). The place where the contour in Fig. 23.10 intersects the imaginary axis is $\mu_i = 1.59$, which is equal to the place where the physical mode intersects the unit circle in Fig. 23.3. Incidentally, the region of stability of LF is on the imaginary axis in the range from $(0, -i)$ to $(0, i)$, and is absolutely unstable with respect to diffusion, etc.

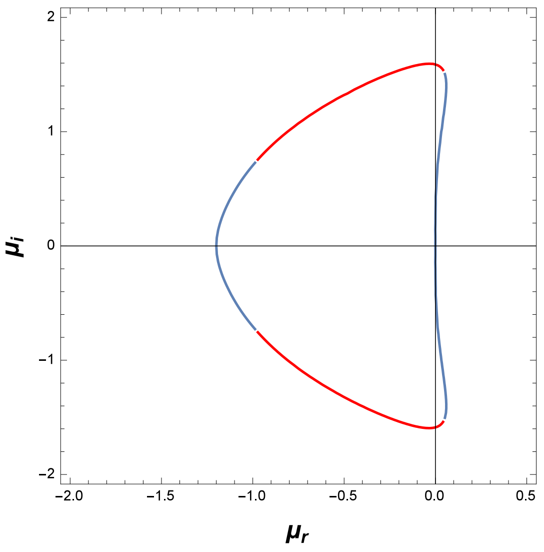

Figure23.10 Contour of $|\lambda| = 1$ on the complex plane of μ. This is equivalent to the curve of LFAM3 in Fig.1(a) in Lemariè et al. (2015). The blue line corresponds to λ^+ and the red lines correspond to λ^-. The region enclosed by the curve is the stable region.

When we draw the region of stability for AB2ϵ and AB3, we get Fig. 23.11. The intersection of the imaginary axes is 0.502 for AB2ϵ versus 0.72 for AB3, which means that AB3 is more stable in the pure wave equation. Since AB2ϵ is more extended toward the real axis, the stability for diffusion is more stable for AB2ϵ. For other schemes, see Fig. 1 of Lemariè et al. (2015).

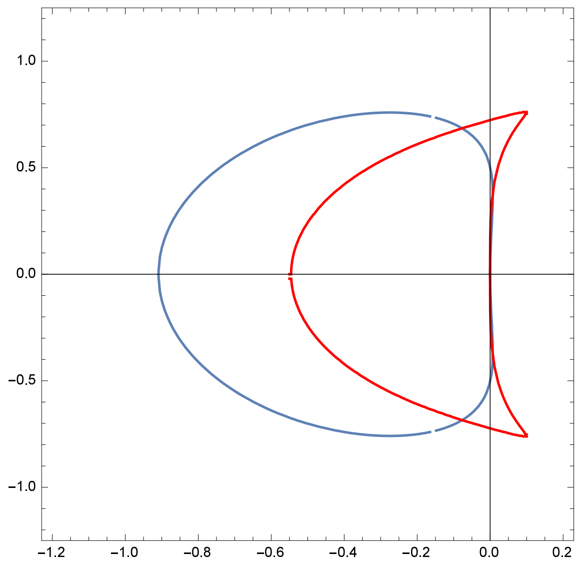

Figure23.11 Contours of $|\lambda| = 1$ on the complex plane of μ, which are equivalent to the curve of AB2ϵ and AB3 in Fig.1(a) in Lemariè et al. (2015). The blue line corresponds to the AB2ϵ method and the red line corresponds to the AB3λ^- method. The regions enclosed by the curves are the stable regions.

Chapter 24

Generalized orthogonal curvilinear coordinate grids

This chapter introduces generalized orthogonal curvilinear coordinates and presents related calculus.

24.1 Outline

A finite volume ocean model on geographic coordinates does not have any problem concerning the South Pole because it does not calculate around the South Pole. However, serious problems arise around the North Pole where the meridian concentrates to one point in the ocean. First, it is necessary to calculate the temporal evolution of the physical quantity in a special way only there, because the relations between U-cells that surround the North Pole and the northernmost T-cell are topologically peculiar. Next, even if a cell doesn't touch the North Pole, its zonal lattice interval is extremely small near the North Pole. Therefore, a short time step for integration is required owing to the limitation of the CFL condition. This limitation is reflected directly in the increased calculation time required. Moreover, when the zonal grid intervals in low latitudes and the Arctic region are extremely different, the arguments about accuracies of numerical schemes and the parameters for diffusion and viscosity operators generally cannot be applied uniformly to a model domain.

The following can be considered to avoid such problems concerning the North Pole. 1) Creating a huge island including the North Pole. The finite-difference calculation in the island is abandoned, and the lateral boundary values are restored to the climatology. 2) Shifting the singular points of the model to a continent or a huge island by changing the model's horizontal grid system. The MRI.COM scheme adopts the latter approach, which is outlined in this section.

Because the MRI.COM code is based on generalized orthogonal coordinates, the geographical latitude (ϕ) and longitude (λ) are not of a great concern for the calculus in the model. However, it is necessary to know the land and sea distribution, sea depth, scale factor, and the Coriolis parameter given as a function of λ and ϕ at every grid point of the model prior to the calculation. We describe the method of generating the orthogonal coordinate grid system in Section 24.2. Using a conformal transformation in the general sense, the functions that describe the relation between model coordinates (μ, ψ) and geographic coordinates (λ, ϕ), $\lambda(\mu, \psi)$, $\phi(\mu, \psi)$, $\mu(\lambda, \phi)$, and $\psi(\lambda, \phi)$, are obtained.

Because an atmospheric boundary condition is given in many cases at grid points in geographic coordinates, it is necessary to prepare tables for converting the surface atmospheric temperature, the wind stress, and so on. To convert a vector quantity, we must remember that the direction of the μ contour differs from that of the λ contour (meridian). The difference is described in Section 24.3. We can use the functions, $(\lambda, \phi) \Longleftrightarrow (\mu, \psi)$, to convert a scalar quantity as well as sea depth and the Coriolis parameter. The total flux that the ocean receives from the atmosphere should be equal to the total flux that the atmosphere gives to the ocean. The method for conserving the total flux is explored in Section 24.4. The vector operation in generalized orthogonal coordinates is concisely described in Section 24.5.

24.2 Generation of orthogonal coordinate system using conformal mapping

We designate the plane that touches the sphere at the North Pole as $\mathbf{S}_\mathrm{N}$. A polar stereographic projection is a conformal transformation in the general sense, so that an orthogonal coordinate system on the sphere is mapped onto an orthogonal coordinate system on $\mathbf{S}_\mathrm{N}$ and the orthogonality is preserved on the reverse transformation (Figure 24.1). Moreover, if $\mathbf{S}_\mathrm{N}$ is assumed to be a complex plane, various conformal transformations can be defined on it. Therefore, applying (i) the polar stereographic projection, (ii) a conformal transformation on the complex plane $\mathbf{S}_\mathrm{N}$, and (iii) the reverse polar stereographic projection to a geographic coordinate grid point (λ, ϕ) on the sphere, an orthogonal coordinate grid point on the sphere can be obtained (Bentzen et al., 1999).

The functions $\mu(\lambda, \phi)$ and $\psi(\lambda, \phi)$ are obtained by the following procedure:

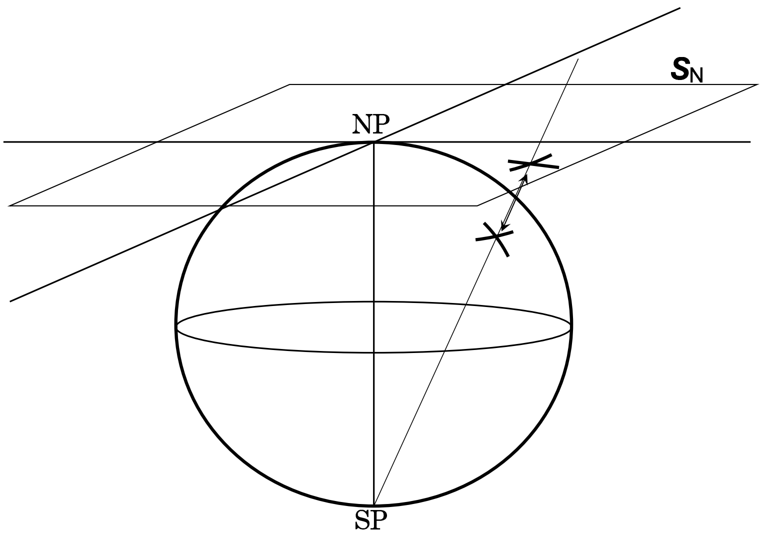

Figure24.1 Schematic illustration of a Polar stereographic projection (a conformal transformation in the general sense between the sphere and $\mathbf{S}_\mathrm{N}$).

1. From a point (λ, ϕ) on the sphere to a point z on $\mathbf{S}_\mathrm{N}$ (polar stereographic projection). Defining colatitude $\phi' = \pi/2 - \phi$,
$$z = \tan\left(\frac{\phi'}{2}\right) e^{i\lambda}, \tag{24.1}$$
where the origin of $\mathbf{S}_\mathrm{N}$ corresponds to $\phi' = 0$ ($\phi = \pi/2$), and the positive part of the real axis corresponds to $\lambda = 0$.
2. Conformal transformation M_C on $\mathbf{S}_\mathrm{N}$:
$$\zeta = M_\mathrm{C}(z). \tag{24.2}$$
3. From a point ζ on $\mathbf{S}_\mathrm{N}$ to a point (μ, ψ) on the sphere (reverse polar stereographic projection).

$$\mu = \arg(\zeta), \tag{24.3}$$
$$\psi' = 2\arctan|\zeta|, \tag{24.4}$$
$$\psi = \pi/2 - \psi'. \tag{24.5}$$

Functions $\lambda(\mu, \psi)$ and $\phi(\mu, \psi)$ are obtained by reversing the above procedure.
Defining $\psi' = \pi/2 - \psi$,
$$\zeta = \tan\left(\frac{\psi'}{2}\right) e^{i\mu}, \tag{24.6}$$

$$z = M_\mathrm{C}^{-1}(\zeta), \tag{24.7}$$

and

$$\lambda = \arg(z), \tag{24.8}$$
$$\phi' = 2\arctan|z|, \tag{24.9}$$
$$\phi = \pi/2 - \phi'. \tag{24.10}$$

Thus, when a model coordinate grid point, $(\mu_0 + \Delta\mu \times (i-1),\ \psi_0 + \Delta\psi \times (j-1))$ is given, we know the geographic position of the point, Coriolis parameter, etc., at once.

Bentzen et al. (1999) used the linear fraction conversion as a conformal transformation on $\mathbf{S}_\mathrm{N}$. That is,

$$\zeta \quad = \quad M_\mathrm{C}(z) \quad = \quad \frac{(z-a)(b-c)}{(c-a)(b-z)}, \tag{24.11}$$

where the three complex numbers a, b, and c expressed by

$$a = \tan\left(\frac{\phi'_a}{2}\right) e^{i\lambda_a}, \quad b = \tan\left(\frac{\phi'_b}{2}\right) e^{i\lambda_b}, \quad c = \tan\left(\frac{\phi'_c}{2}\right) e^{i\lambda_c}, \tag{24.12}$$

correspond to the three geographic coordinate grid points (λ_a, ϕ_a), (λ_b, ϕ_b), and (λ_c, ϕ_c), which are mapped to the model coordinate grid points, $(\mu, \psi) = (0, \pi/2)$, $(0, -\pi/2)$, $(0, 0)$, respectively. Therefore, the singular point $(\mu, \psi) = (0, \pi/2)$ in the model calculation can be put on Greenland, by setting (λ_a, ϕ_a) at 75°N and 40°W. If `TRIPOLAR` or `JOT` option is specified instead of `SPHERICAL`, then two singular points: $(\mu, \psi) = (0, \pi/2)$ and $(0, -\pi/2)$ can be put on arbitrary land locations by suitably setting (λ_a, ϕ_a) and (λ_b, ϕ_b), which are model parameters (`north_pole_lon`, `north_pole_lat`, `south_pole_lon`, and `south_pole_lat` in degree) that should be specified in namelist `nml_poles` (Table 3.3).

When `TRIPOLAR` option is specified, the parameters are set to $\phi_a = \phi_b =$ 64°N, $\lambda_a =$ 80°E, and $\lambda_b =$ 100°W. The transformed grids are used for the region north of 64°N, and geographic coordinates are used for the region south of 64°N. This tripolar coordinate system can express the Arctic Sea with a higher resolution than the Southern Ocean. The adoption of geographic coordinates south of 64°N enables us to do the assimilation and analysis with relative ease (Figure 24.2).

When `JOT` option is specified, the Joukowski conversion is used as a conformal transformation on $\mathbf{S}_\mathrm{N}$. That is,

$$z = M_C^{-1}(\zeta) = \left(\zeta + \frac{\psi_0'^2}{\zeta}\right) e^{i\mu_0}. \tag{24.13}$$

This Joukowski conversion maps the area outside the circle with a radius of ψ'_0 centered at the origin of the ζ-plane to the whole domain of the z-plane, and rotates it by μ_0. The left panel of Figure 24.2 presents an example where the coordinate system is created by setting ψ'_0 to 20° and μ_0 to 80°. Because there is no discontinuity of grid spacing in this coordinate system, the singular points can be put at various positions. For instance, the singular point on the North American side can be put on the Labrador Peninsula or in Greenland.

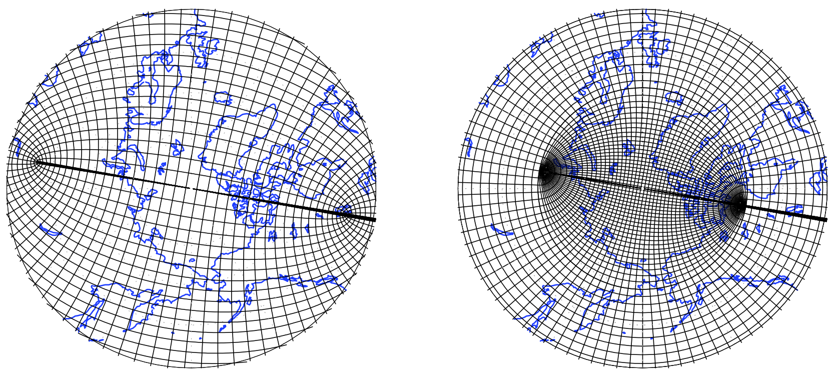

Figure24.2 Model coordinate grid arrangement in the Arctic sea. Left: Grid system made through the Joukowski conversion (`JOT`). Right: Combination of the coordinate systems made through the linear fraction conversion and conventional geographic coordinates (`TRIPOLAR`).

Functions $\lambda(\mu, \psi)$ and $\phi(\mu, \psi)$ are defined as subroutine `mp2lp`, and functions $\mu(\lambda, \phi)$ and $\psi(\lambda, \phi)$ are defined as subroutine `lp2mp`. Module programs `trnsfrm.{spherical, moebius, tripolar, jot}.F90` contain these internal subroutines. These functions, especially `mp2lp`, are frequently used when the topography and the surface boundary condition are made before starting the main integration of model.

24.3 Rotation of vector

A vector expressed in geographic coordinates (λ, ϕ) should be rotated when observed from model coordinates (μ, ψ). Taking advantage of the local orthogonality of coordinate axes, the angle (α) is obtained as an angle at which the meridian

($\lambda = \lambda_0$) of geographic coordinates intersects that of the model coordinates ($\mu = \mu_0$) at a certain point. First, we set

$$z = f(\zeta), \quad z = x + iy, \quad \zeta = u + iv, \tag{24.14}$$

$$f'(\zeta) \quad = \quad \frac{\partial x}{\partial u} + i\frac{\partial y}{\partial u} \quad = \quad \frac{\partial y}{\partial v} - i\frac{\partial x}{\partial v}. \tag{24.15}$$

At a certain point $z_0 = f(\zeta_0)$ on geographic coordinates, the angle θ at which a curve $v = v_0$ meets a straight line $y = y_0$ is given by

$$\tan\theta = \left[\frac{\partial y}{\partial x}\right]_{v_0} = \left[\frac{\partial y}{\partial u}\Big/\frac{\partial x}{\partial u}\right]_{v_0},$$

then (see Figure 24.3),

$$\theta = \arg(f'(\zeta_0)). \tag{24.16}$$

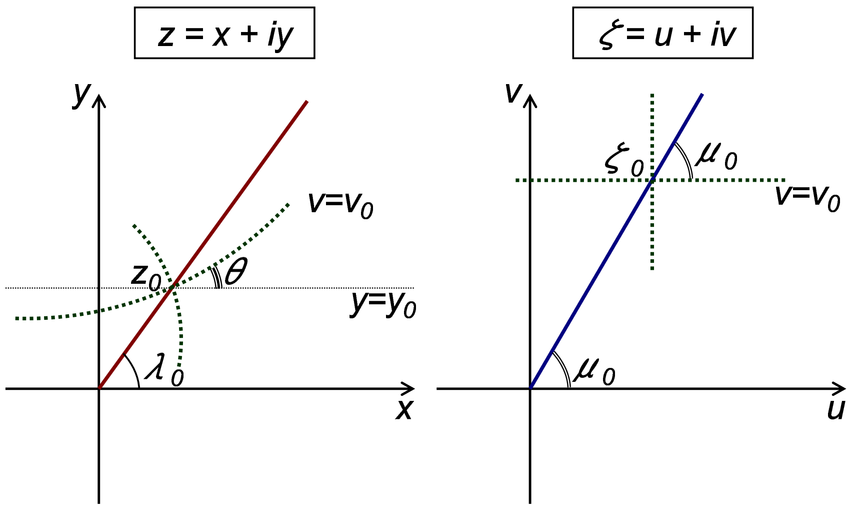

Figure24.3 A meridian (red) in geographic coordinates (λ, ϕ) (left) and a meridian (blue) in model coordinates (μ, ψ) (right).

Assuming $\lambda = \arg(z)$ and $\mu = \arg(\zeta)$, at point ζ_0 the straight line ($v = v_0$) meets the straight line ($\mu = \mu_0$) at the angle $-\mu_0\,(\mu_0 \to v_0)$, and at point z_0 the meridian ($\lambda = \lambda_0$) meets the curve ($v = v_0$) at the angle $\lambda_0 - \theta\,(v_0 \to \lambda_0)$. The meridian ($\lambda = \lambda_0$) in λ-ϕ coordinates meets the line ($\mu = \mu_0$) in μ-ψ coordinates at angle α given as follows:

$$\begin{aligned}\alpha &= -\mu_0 + \lambda_0 - \theta \\ &= \lambda_0 - \mu_0 - \arg(f'(\zeta_0)).\end{aligned} \tag{24.17}$$

Subroutine `rot_mp2lp` defined in `trnsfrm.{spherical,moebius,tripolar,jot}.F90` returns $(\cos\alpha, \sin\alpha)$ at a specified grid point of the model. A wind stress vector (τ_x, τ_y) in geographic coordinates should appear in the model ocean described in the μ-ψ coordinate system as $(\tau_x\cos\alpha - \tau_y\sin\alpha, \tau_x\sin\alpha + \tau_y\cos\alpha)$.

24.4 Mapping a quantity from geographic coordinates to transformed coordinates

We consider a method to receive a quantity $G_{I,J}$ given at the geographic coordinate grids (I, J) as the quantity $H_{M,N}$ at the model coordinate grids (M, N) (Figure 24.4). The quantities are wind stress components after the vector rotation, precipitation per unit area, sea surface atmospheric temperature, and so on. In addition, the average depth at a model grid point can also be calculated by the following method because bottom topography (depths of sea floor) is usually given in geographic coordinates.

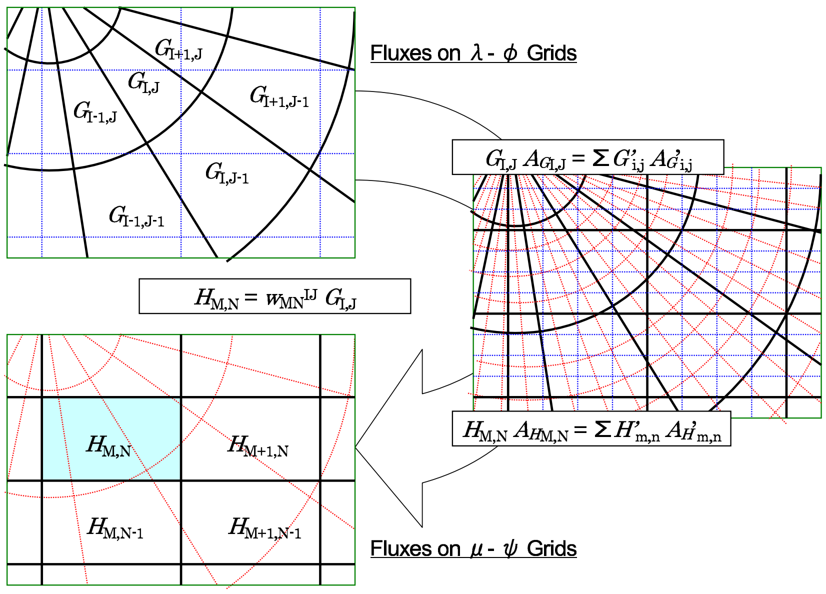

Figure24.4 Grids (I, J) and (M, N) subdivided into finer grids (i, j) and (m, n).

Grids (I, J) and (M, N) are suitably subdivided into finer grids (i, j) and (m, n). We call these filter grids. A quantity $G'_{i,j}$ is assumed to be homogeneously distributed in the geographic filter grids (i, j) covered by grid (I, J),

$$G'_{i,j} = G_{I,J}.$$

Assume the quantity at a model filter grid $H'_{m,n}$ is equal to that at the nearest geographic filter grid,

$$H'_{m,n} = G'_{i(m,n),j(m,n)}.$$

The quantity at model grid (M, N) is obtained as the area-weighted average:

$$H_{M,N} = \frac{1}{A_{H\,M,N}} \sum_{m,n} A_{H'm,n} H'_{m,n}, \tag{24.18}$$

where $A_{H\,M,N}$ is the area of model grid and $A_{H'm,n}$ is the area of model filter grid.

When the grid intervals of geographic filter grid (i, j) and model filter grid (m, n) are extremely small, the total quantity (flux) received on the model grids (M, N) is equal to the total quantity (flux) given by the geographic grids (I, J). The relation between the quantity in the geographic grids and that in the model grids is defined by weight w,

$$H_{M,N} = \sum_{I,J} w(M, N, I, J) G_{I,J}. \tag{24.19}$$

How is the quantity converted in an actual calculation in the model?

1. When the strict conservation of quantity (flux) is necessary:
   Fresh water is not permitted to be generated or vanish at the surface boundary in a run using an atmosphere-ocean coupled model, for instance. In this case, $w(M, N, I, J)$ is prepared beforehand, and the flux is passed from the atmosphere through equation (24.19) to the ocean. The resolution of the filter grid need not be extremely fine, provided that every geographic filter grid is linked to more-than-zero model filter grids and

$$\sum_{I,J} A_{G\,I,J} = \sum_{i,j} A'_{G'i,j} = \sum_{m,n} A'_{H'm,n} = \sum_{M,N} A_{H\,M,N}.$$

2. When conservation need not be guaranteed:
   When the ocean model is driven by the surface boundary condition based on atmospheric re-analysis data, the amount of fresh water entering the sea as precipitation and river discharge is not equal to that drawn from the ocean

through evaporation and sublimation. Therefore, the global sea surface height rises or descends during years of integration. It is not very important to pursue complete conservation of fresh-water under this condition. In such a case, the flux at a model grid point can be prepared beforehand using equation (24.19), to avoid the time-consuming flux conversions in the model calculation.

24.5 Vector operation and differentiation in generalized orthogonal coordinates

To formulate the model equations, we have to know the vector operation and differentiation in generalized orthogonal coordinates. Some basic formulae used in formulating primitive equations are presented here.

The line element vector $\delta\mathbf{x}$ at a certain point (μ, ψ, r) in an arbitrary general orthogonal coordinate system is expressed as

$$\delta\mathbf{x} = h_\mu \delta\mu \mathbf{e}_\mu + h_\psi \delta\psi \mathbf{e}_\psi + h_r \delta r \mathbf{e}_r, \tag{24.20}$$

where basis vectors $\mathbf{e}_\mu$, $\mathbf{e}_\psi$, and $\mathbf{e}_r$ are mutually orthogonal unit vectors, and h_μ, h_ψ, and h_r are scale factors.

Defining

$$\nabla = \frac{\mathbf{e}_\mu}{h_\mu}\frac{\partial}{\partial\mu} + \frac{\mathbf{e}_\psi}{h_\psi}\frac{\partial}{\partial\psi} + \frac{\mathbf{e}_r}{h_r}\frac{\partial}{\partial r}, \tag{24.21}$$

the gradient of scalar $A(\mu, \psi, r)$ is

$$\nabla A = \frac{\mathbf{e}_\mu}{h_\mu}\frac{\partial A}{\partial\mu} + \frac{\mathbf{e}_\psi}{h_\psi}\frac{\partial A}{\partial\psi} + \frac{\mathbf{e}_r}{h_r}\frac{\partial A}{\partial r}, \tag{24.22}$$

and the divergence of vector $\mathbf{A} = A_\mu \mathbf{e}_\mu + A_\psi \mathbf{e}_\psi + A_r \mathbf{e}_r$ is

$$\nabla \cdot \mathbf{A} = \frac{1}{h_\mu h_\psi h_r}\left[\frac{\partial(h_\psi h_r A_\mu)}{\partial\mu} + \frac{\partial(h_r h_\mu A_\psi)}{\partial\psi} + \frac{\partial(h_\mu h_\psi A_r)}{\partial r}\right]. \tag{24.23}$$

The r component of curl$\mathbf{A}$ is

$$\frac{1}{h_\mu h_\psi}\left[\frac{\partial(h_\psi A_\psi)}{\partial\mu} - \frac{\partial(h_\mu A_\mu)}{\partial\psi}\right]. \tag{24.24}$$

The calculation of velocity advection includes $(\mathbf{a}\cdot\nabla)\mathbf{A}$, where $\mathbf{a}$ is an arbitrary vector ($\mathbf{a} = a_\mu \mathbf{e}_\mu + a_\psi \mathbf{e}_\psi + a_r \mathbf{e}_r$).

The μ component of $(\mathbf{a}\cdot\nabla)\mathbf{A}$ is

$$\mathbf{a}\cdot\nabla A_\mu + \frac{A_\psi}{h_\mu h_\psi}\left(a_\mu \frac{\partial h_\mu}{\partial\psi} - a_\psi \frac{\partial h_\psi}{\partial\mu}\right) + \frac{A_r}{h_r h_\mu}\left(a_\mu \frac{\partial h_\mu}{\partial r} - a_r \frac{\partial h_r}{\partial\mu}\right). \tag{24.25}$$

The second and third terms are so-called "metric" terms in the equation of motion in spherical coordinates.

These expressions in spherical coordinates (λ, ϕ, r) are shown next. Defining longitude λ, latitude ϕ, and radius of the earth r, scale factors are $h_\lambda = r\cos\phi$, $h_\phi = r$, and $h_r = 1$.

Velocity vector $\mathbf{v}$ is

$$\mathbf{v} = u\mathbf{e}_\lambda + v\mathbf{e}_\phi + w\mathbf{e}_r, \tag{24.26}$$

where $\mathbf{e}_\lambda$, $\mathbf{e}_\phi$, and $\mathbf{e}_r$ are the eastward, northward, and upward unit vectors, respectively, and $(u, v, w) = (r\cos\phi\dot{\lambda}, r\dot{\phi}, \dot{r})$.

The gradient of scalar function $A(\lambda, \phi, r)$ is,

$$\nabla A = \frac{\mathbf{e}_\lambda}{r\cos\phi}\frac{\partial A}{\partial\lambda} + \frac{\mathbf{e}_\phi}{r}\frac{\partial A}{\partial\phi} + \mathbf{e}_r\frac{\partial A}{\partial r}, \tag{24.27}$$

where

$$\nabla = \frac{\mathbf{e}_\lambda}{r\cos\phi}\frac{\partial}{\partial\lambda} + \frac{\mathbf{e}_\phi}{r}\frac{\partial}{\partial\phi} + \mathbf{e}_r\frac{\partial}{\partial r}. \tag{24.28}$$

For vector $\mathbf{A} = A_\lambda \mathbf{e}_\lambda + A_\phi \mathbf{e}_\phi + A_r \mathbf{e}_r$, the divergence is

$$\nabla\cdot\mathbf{A} = \frac{1}{r\cos\phi}\left[\frac{\partial A_\lambda}{\partial\lambda} + \frac{\partial(\cos\phi A_\phi)}{\partial\phi}\right] + \frac{\partial(r^2 A_r)}{r^2\partial r}. \tag{24.29}$$

and the r component of curl$\mathbf{A}$ is

$$[\mathrm{curl}\mathbf{A}]_r = \frac{1}{r\cos\phi}\left[\frac{\partial A_\phi}{\partial\lambda} - \frac{\partial(\cos\phi A_\lambda)}{\partial\phi}\right]. \tag{24.30}$$

The λ component of $(\mathbf{a} \cdot \nabla)\mathbf{A}$ is

$$[(\mathbf{a} \cdot \nabla)\mathbf{A}]_\lambda = \mathbf{a} \cdot \nabla A_\lambda - \frac{A_\phi a_\lambda \tan\phi}{r} + \frac{A_r a_\lambda}{r}. \tag{24.31}$$

The Coriolis force in generalized orthogonal coordinates (μ, ψ, r) is given as

$$2\Omega \times \mathbf{v} = (2\Omega_\psi w - 2\Omega_r v)\mathbf{e}_\mu + (2\Omega_r u - 2\Omega_\mu w)\mathbf{e}_\psi + (2\Omega_\mu v - 2\Omega_\psi u)\mathbf{e}_r, \tag{24.32}$$

where $\Omega = \Omega_\mu \mathbf{e}_\mu + \Omega_\psi \mathbf{e}_\psi + \Omega_r \mathbf{e}_r$ is the rotation vector of the Earth, and $\mathbf{v} = u\mathbf{e}_\mu + v\mathbf{e}_\psi + w\mathbf{e}_r$ is the velocity vector. We designate $f_\mu = 2\Omega_\mu$, $f_\psi = 2\Omega_\psi$, and $f = f_r = 2\Omega_r$ in Chapter 2. The rotation vector of the Earth is $(\Omega_\lambda, \Omega_\phi, \Omega_r) = (0, \Omega\cos\phi, \Omega\sin\phi)$ in geographic coordinates (λ, ϕ, r).

Chapter 25

User's Guide

This chapter briefly explains the procedures needed to run MRI.COM. The description in this chapter is based on MRI.COM version 5.0 and some contents presented in this chapter may not be used for the latest version. It is recommended that users refer to `README_First.md`, `README_Options.md`, `README_Namelist.md`, `README_Monitor.md`, and `README_Restart.md` in the `docs/` directory when setting up a model.

The minimal information to prepare, run, and post-process is presented in this chapter in the following order:

- **Model setup:** User defined parameter files and compilation (Section 25.1 and Table 25.1).
- **Input data:** Grid spacing, topography, and surface forcing etc., to be read at run time (Section 25.2 and Table 25.2).
- **Restart file:** Explanation of restart files (Section 25.3).
- **Execution:** Explanation of the runtime parameters that control time-integration (Section 25.4).
- **Post process:** A description of monitor files (Section 25.5).

Note that cgs units are employed to express physical values in the model.

We are developing a comprehensive package of tools "MRI.COM eXecution Environment (MXE)" that aggregates programs for preprocessing, execution, postprocessing, and analysis of MRI.COM experiments. This is briefly explained in Section 25.6. Appendix (Sec. 25.7) contains website information and a list of model compilation options.

25.1 Model setup

25.1.1 Model configuration file (configure.in)

This section describes the procedures necessary for setting up the model and compiling its programs. First, prepare `configure.in` that contains the information about model options and grid size. This is needed for compilation. Additional parameters are required for some particular model options. Those are listed in Table 25.1 (see also `README_Options.md`). Following is an example of `configure.in`.

An example `configure.in` for Global tripolar $1^\circ \times 0.5^\circ$ grid model

```
DEFAULT_OPTIONS="IDEALAGE ICE SIDYN CALALBSI SMAGOR VIS9P DIFAJS GLS VVDIMP
 SOMADVEC ISOPYCNAL HFLUX TAUBULK WFLUX RUNOFF SFLUXR BULKNCAR
 CYCLIC BBL TRIPOLAR PARALLEL"
NAME_MODEL='GLOBAL'
IMUT=364
JMUT=368
KM=51
NPARTX=8
NPARTY=4
NUM_ICECAT=5
NUMTRC_P=1
```

Table25.1 Model parameters to be set in configure.in

option name	variable name	description
always required	IMUT, JMUT, KM	zonal/meridional/vertical grid number
	NAME_MODEL	name of the model (default = "tmOGCM")
	NSFMRGN	the number of side-boundary ghost cells to reduce the communication cost in parallel computation (see Ishizaki and Ishikawa, 2006)
PARALLEL	NPARTX, NPARTY	the number of zonally/meridionally partitioned region for a computation using parallel processors: the number of parallel processes should be NPARTX $\times$ NPARTY
passive tracers	NUMTRC_P	only when any passive tracer is calculated (NUMTRC_P $\geq$ 1)
ICE	NUM_ICECAT	the number of thickness categories of sea ice

25.1.2 Compilation of the model

A standard compiling script is prepared as `compile.sh` in the `src/` directory. The part depending on the system (OS, Fortran compiler, and compiler option) are found in `Machines/` directory.

To compile the programs, execute `compile.sh`. The script `compile.sh` creates `param.F90`, `Icecat/ice_param.F90`, and `Makefile` from `configure.in` by running `configure`, and then executes the command `make` to create the executable file `ogcm`. The environment variables for compilation are set in `configure` using the options prescribed in `configure.in`. If `configure.in` is newer than `param.F90`, a new `param.F90` is created based on the parameter values defined in `configure.in`.

The program files that should be compiled are automatically selected according to the descriptions of the relationships in `Makefile`, but users should be careful since it might not be perfect. The compilation should be carried out after executing `make clean`, when any compile option in `configure.in` is changed.

25.2 Preparation of input data files for execution

Data files listed on Table 25.2 must be prepared, according to user's specification of compile options and runtime parameters. See Section 25.3 for how to handle restart files.

Table25.2 Main input data files and their related program files. Here, *name_model* represents the specific name given as NAME_MODEL in configure.in.

subject	file name specified in (NAMELIST.*name_model*)
runtime parameters	NAMELIST.*name_model*
specification about monitoring	NAMELIST.*name_model*.MONITOR
variable horizontal grid spacing	file_dxdy_tbox_deg (nml_horz_grid)
variable vertical grid spacing	file_dz_cm (nml_vert_grid)
grid cell area and line elements	file_scale (nml_grid_scale)
topography	file_topo (nml_topo)
reference data for tracers	trcref(_surf)_conf%file_data
restoring coefficient for tracers	rstcoef(_surf)_conf%file_data
surface forcing	file_data (nml_force_data)
surface forcing grid	file_data_grid (nml_force_data)
restart files	file_base (nml_restart)

25.2.1 Grid spacing and cell area

Details on how to specify grid information and how to prepare the necessary data is given in Section 3.6. The grid spacing data file should be prepared for each of the horizontal (`file_dxdy_tbox_deg`) or vertical (`file_dz_cm`) directions when variable grid spacing is used for that direction. The units are in degrees for the horizontal and in cm for the vertical.

When the model grid points are defined on the basis of general orthogonal coordinates, the quarter cell area and line elements should be prepared. The units are in cgs. It is read from the file `file_scale`.

When spherical coordinates are used (`SPHERICAL` option), e.g., the grids are defined on geographical latitude and longitude, the grid information is analytically calculated in the model, and the file `file_scale` is not necessary.

25.2.2 Topography

Land-sea distribution and sea-floor topography should be given by the topography data file `file_topo`. The topographic data consist of the 4-byte integer array `HO4(imut, jmut)` that contains the sea floor depths of the velocity grid points (in cm) and the 4-byte integer array `EXNN(imut, jmut)` that contains its corresponding vertical level. They should be written unformatted and sequentially as follows:

Format of topographic data (`file_topo`)

```
integer(4) :: ho4(imut,jmut),exnn(imut,jmut)
open(unit=nu,file=file_topo)
write(unit=nu) ho4, exnn
```

An example of the topography for global 1° × 0.5° model is shown in Figure 25.1. In creating a model topography, especially for a low-resolution model, the user should be careful that the important gateways for the ocean circulation be kept open and that the land blocking the ocean circulation be kept closed.

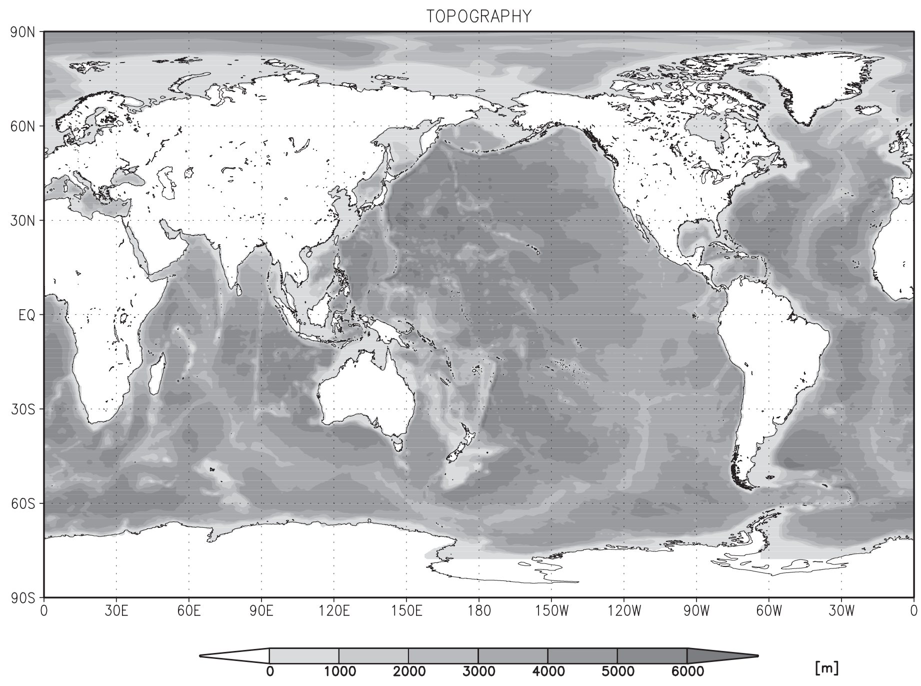

Figure25.1 Example of ocean model topography (global 1° × 0.5° grid model).

25.2.3 Reference data and restoring coefficient for tracers

For each tracer, the integration may be started from an initial state based on the reference data and the tracer values may be restored to the reference data during the integration. To do this, tracer reference values and restoring coefficients should be prepared. See Chapter 13 for how to prepare these data.

25.2.4 Surface forcing data

See Chapter 14 for how to prepare surface forcing data. The surface forcing data are read at a uniform time interval. A leap year is set according to an optional namelist, `nml_calendar` (Table 25.10). Climatological data may be used repeatedly.

The following data files should be prepared according to the chosen model options. Each file is opened only once at the beginning of run time and thus should contain all the data needed for that run.

Table25.3 Surface data to be read from namelist nml_force_data.

	name	units	usage
X-ward wind stress	U-wind	dyn cm^{-2}	if not TAUBULK
Y-ward wind stress	V-wind	dyn cm^{-2}	if not TAUBULK
X-ward wind speed	U-wind	$cm\,s^{-1}$	if TAUBULK
Y-ward wind speed	V-wind	$cm\,s^{-1}$	if TAUBULK
Downward shortwave radiation	ShortWave	$erg\,s^{-1}\,cm^{-2} = 10^{-3}\,W\,m^{-2}$	
Downward longwave radiation	LongWave	$erg\,s^{-1}\,cm^{-2}$	
Surface air temperature	TempAir	°C	
Surface air specific humidity	SphAir	1	
Scalar wind speed	ScalarWind	$cm\,s^{-1}$	unnecessary if TAUBULK
Sea level pressure	SeaLevelPressure	hPa	also available for SLP
precipitation	Precipitation	$g\,s^{-1}\,cm^{-2}$	WFLUX
river discharge	RiverDischargeRate	$g\,s^{-1}\,cm^{-2}$	RUNOFF
Sea ice area fraction	IceConcentrationClimatology	1	ICECLIM

25.3 Restart file

A set of restart files provides an instantaneous state necessary to resume a model integration. This section explains how to handle restart files.

25.3.1 Restart for the ocean model

Restart files for the ocean model are specified using a common namelist block, nml_restart, in the same manner as history outputs. The namelists must be written in NAMELIST.*name_model* (default) as follows:

An example namelist for temperature restart files

```
&nml_restart
   name      = 'temperature',
   file_base = 'result/rs_t',
   (interval_step = 0,)
/
```

where name is a variable name to be input/output (listed in Table 25.4), and file_base specifies the basename of the restart files. This information is shared by both input and output files. In the above example, the name of a restart file for temperature is result/rs_t.YYYYMMDDHHMMSS. A suffix indicating the date and time of data is always added to the basename. (Namlist block nmlrs_ in MRI.COMv4 has been removed.)

Table 25.4 lists available names in nml_restart (name is not case sensitive). Required elements depend on model options. See docs/README_Restart.md for more information. An example for a set of biogeochemical tracers is also shown in Section 19.6.

Table25.4 Restart variables and their namelist block that specifies their restart file attributes.

variable name	model options	namelist in MRI.COMv4
Potential temperature		nmlrs_t
Salinity		nmlrs_s
X Velocity		nmlrs_u
Y Velocity		nmlrs_v

Continued on next page

Table 25.4 – continued from previous page

variable name	model options	namelist in MRI.COMv4
Sea Surface Height		nmlrs_ssh
X Transport		nmlrs_uml
Y Transport		nmlrs_vml
X diffusion flux for SSH		nmlrs_ssh_dflx_x
Y diffusion flux for SSH		nmlrs_ssh_dflx_y
Global Mean Density and Pressure	required if CALPP	nmlrs_density
Vertical Tracer Diffusivity	VVDIMP	nmlrs_vmix_avd
Vertical Momentum Diffusivity	VVDIMP	nmlrs_vmix_avm
passive tracers (see Section 19.6)	NUMTRC_P > 0	nmlrs_ptrc
Tidal X Transport	TIDE	nmlrs_tide_um
Tidal Y Transport	TIDE	nmlrs_tide_vm
Tidal Sea Surface Height	TIDE	nmlrs_tide_ssh
M-Y eddy kinetic energy diffusion	MELYAM	nmlrs_my_avq
M-Y turbulent velocity scale	MELYAM	nmlrs_my_q
M-Y length scale	MELYAM	nmlrs_my_alo
Noh-Kim eddy kinetic energy diffusion	NOHKIM	nmlrs_nk_avq
Noh-Kim eddy kinetic energy	NOHKIM	nmlrs_nk_eb
GLS eddy kinetic energy diffusion	GLS	nmlrs_gls_avk
GLS generic variable diffusion	GLS	nmlrs_gls_avp
GLS eddy kinetic energy	GLS	nmlrs_gls_eke
GLS generic variable	GLS	nmlrs_gls_psi
GLS length scale	GLS	nmlrs_gls_alo
(see Section 25.3.2)	SOMADVEC and adv_scheme%name = "som" in nml_tracer_data for any tracer	nml_somadv
Sea ice	ICE	nml_seaice_rst_in

Restart files are saved in a Fortran direct-access format, which can be read by a following program.

Program to read a restart file of the ocean model

```
character(14)         :: date = '20010101000000'  !- for 0:00z1JAN2001
real(8)               :: d(imut,jmut,km)
integer(4),parameter :: nu = 10 ! device number

open(nu, file='result/rs_t.'//date, form='unformatted', &
      & access='direct',recl=imut*jmut*km*8)
read(nu,rec=1) d
close(nu)
```

The file endian is that of the computer where the model runs. There are support tools to visualize restart files in the directory, tools/MK_GRADS. Please follow the instructions given by tools/MK_GRADS/00README.txt. The MXE package (Section 25.6) also has tools for visualization in the directory, postp/.

25.3.2 Restart for the SOM advection scheme

When the SOM tracer advection scheme is used (SOMADVEC), the model may read and write additional 9 restart files of the 2nd order moments for each tracer. Their names are BASENAME_SOM_SS.YYYYMMDDHHMMSS, where BASENAME indicates a character string specified by file_base in namelist nml_restart, SS elements of the moments (9 types), and the suffix for the date and time. The format of each file is the same as the tracer restart files. See Section 10.5 for the SOM scheme, and nml_tracer_data in docs/README_Namelist.md to control the SOM restart input/output. (Naming of the SOM restart files was changed in MRI.COMv5. Use tools/ver46to47/SOMRESTART/ to rename SOM restart files.)

25.3.3 Restart for the sea ice model

To handle restart files of the sea ice model, a namelist block, `nml_restart` with `name` = "Sea ice", must be specified. (`nml_seaice_rst_in` and `nml_seaice_rst_out` are obsolete.) The restart file contains a number of variables needed to restart the sea ice model, and can be read by the following program.

Program to read restart for the sea ice model

```
integer(4), parameter :: ncat = 5 ! number of thickness categories
! ice concentration
real(8) :: aicen (1:imut,1:jmut,0:ncat), a0iceo(imut,jmut)
! ice thickness
real(8) :: hicen (1:imut,1:jmut,0:ncat), hiceo (imut,jmut)
! averaged sea-ice thickness
real(8) :: hin   (1:imut,1:jmut,0:ncat), hi    (imut,jmut)
! snow depth
real(8) :: hsnwn (1:imut,1:jmut,0:ncat), hsnwo (imut,jmut)
! averaged snow thickness
real(8) :: hsn   (1:imut,1:jmut,0:ncat), hsnw  (imut,jmut)
! ice surface temperature
real(8) :: tsfcin(1:imut,1:jmut,0:ncat), tsfci (imut,jmut)
! ice temperature
real(8) :: t1icen(1:imut,1:jmut,0:ncat)
! sea surface skin temperature
real(8) :: t0icen(1:imut,1:jmut,0:ncat)
! sea surface skin salinity
real(8) :: s0n   (1:imut,1:jmut,0:ncat)
! skin temperature beneath the sea ice
real(8) :: t0iceo(imut,jmut)
! skin temperature in the open leads
real(8) :: t0icel(imut,jmut)
! stress tensor
real(8) :: sigma1(imut,jmut), sigma2(imut,jmut), sigma3(imut,jmut)

integer(4),parameter :: nu = 10 ! device number
integer(4)          :: i = 0, m

open(nu, file='result/rs_ice', form='unformatted', &
  & access='direct', recl=imut*jmut*8)

do m = 0, ncat ; i = i + 1 ; read(nu,rec=i) aicen(1:imut,1:jmut,m) ; enddo
do m = 0, ncat ; i = i + 1 ; read(nu,rec=i) hin  (1:imut,1:jmut,m) ; enddo
do m = 0, ncat ; i = i + 1 ; read(nu,rec=i) hsn  (1:imut,1:jmut,m) ; enddo
do m = 0, ncat ; i = i + 1 ; read(nu,rec=i) hicen(1:imut,1:jmut,m) ; enddo
do m = 0, ncat ; i = i + 1 ; read(nu,rec=i) hsnwn(1:imut,1:jmut,m) ; enddo
do m = 0, ncat ; i = i + 1 ; read(nu,rec=i) tsfcin(1:imut,1:jmut,m) ; enddo
do m = 0, ncat ; i = i + 1 ; read(nu,rec=i) t1icen(1:imut,1:jmut,m) ; enddo
do m = 0, ncat ; i = i + 1 ; read(nu,rec=i) t0icen(1:imut,1:jmut,m) ; enddo
do m = 0, ncat ; i = i + 1 ; read(nu,rec=i) s0n(1:imut,1:jmut,m) ; end do

i = i + 1 ; read(nu,rec=i) a0iceo(1:imut,1:jmut)
i = i + 1 ; read(nu,rec=i) hi(1:imut,1:jmut)
i = i + 1 ; read(nu,rec=i) hsnw(1:imut,1:jmut)
i = i + 1 ; read(nu,rec=i) hiceo(1:imut,1:jmut)
i = i + 1 ; read(nu,rec=i) hsnwo(1:imut,1:jmut)
i = i + 1 ; read(nu,rec=i) uice(1:imut,1:jmut)
i = i + 1 ; read(nu,rec=i) vice(1:imut,1:jmut)
i = i + 1 ; read(nu,rec=i) sigma1(1:imut,1:jmut)
i = i + 1 ; read(nu,rec=i) sigma2(1:imut,1:jmut)
i = i + 1 ; read(nu,rec=i) sigma3(1:imut,1:jmut)

close(nu)
```

25.3.4 Control of input and output at run time

a. Input

Input of restart files at run time is controlled by namelist listed on Table 25.5. By setting `l_rst_in = .true.` in `nml_run_ini_state`, all necessary elements are read from restart files. However, the specification by `l_rst_in` may be overridden for some elements by specifying namelists dedicated to those elements. Note that model tries to read restart files for all passive tracers, thus namelist `nmlrs_ptrc` must be always specified appropriately.

Table25.5 Namelist blocks that control input of restart files.

namelist block	variable name	Usage
`nml_run_ini_state`	`l_rst_in`	`.true.` : read restart files and resume integration `.false.` (default): initial state is set internally No motion (u,v,ssh=0). Initial condition for temperature and salinity are deteremined by `nml_tracer_run`
`nml_barotropic_run`	`l_rst_barotropic_in`	default = `l_rst_in`
`nml_baroclinic_run`	`l_rst_baroclinic_in`	default = `l_rst_in`
`nml_tracer_run`	`l_rst_tracer_in`	default = `l_rst_in` applicable to only temperature and salinity
`nml_vmix_run`	`l_rst_vmix_in`	default = `l_rst_in`
`nml_tide_run`	`l_rst_tide_in`	default = `l_rst_in`
`nml_melyam_run`	`l_rst_melyam_in`	default = `l_rst_in`
`nml_nohkim_run`	`l_rst_nohkim_in`	default = `l_rst_in`
`nml_gls_run`	`l_rst_gls_in`	default = `l_rst_in`
`nml_density_run`	`l_rst_density_in`	default = `l_rst_in`
`nml_seaice_run`	`l_rst_seaice_in`	default = `.false.`
`nml_tracer_data`	`adv_scheme%lrstin_som`	= `.true.` to read restart of SOM scheme
`nml_tracer_data`	`adv_scheme%lrstout_som`	= `.true.` to write restart of SOM scheme

b. Output

Model always try to write restart files of the oceanic part according to the specification by the namelists listed on Table 25.4. Users have to specify those namelists appropriately to obtain restart files and terminate the model normally. Output from the SOM scheme may be suppressed if `adv_scheme%lrstout_som` = `.false.` in `nml_tracer_data`.

25.4 Execution

To run a model, a shell script that handles input/output files, executes the compiled binary `ogcm`, and post-processes is usually prepared. The following two namelist files have to be given.

- `NAMELIST.`*name_model* gives runtime parameters. See `docs/README_Namelist.md` for details.
- `NAMELIST.`*name_model*`.MONITOR` specifies sampling. See `docs/README_Monitor.md` for details.

Runtime parameters relevant to time-integration are listed on Tables 25.6 through 25.11. Other parameters are explained in corresponding chapters.

Table25.6 Namelist `nml_time_step`.

variable name	units	description	usage
`dt_sec`	sec, real(8)	unit time interval for equation of motion and tracer	required
`alpha_bryan_1984`	real(8)	acceleration parameter α of Bryan (1984). See Section 2.3.1.	default = 1
`l_monitor_time`	logical	time step monitor is written to standard output ()	default = `.true.`

Table25.7 Namelist `nml_run_period`.

variable name	units	description	usage
`nstep_total`	integer	total number of time step of this run	required

Table25.8 Namelist `nml_exp_start` that specifies start time of the whole experiment.

variable name	units	description	usage
`year`	integer(4)	year	default = −999
`month`	integer(4)	month	default = 1
`day`	integer(4)	day	default = 1
`hour`	integer(4)	hour	default = 0
`minute`	integer(4)	minute	default = 0
`second`	integer(4)	second	default = 0

Table25.9 Namelist `nml_run_ini` that specifies start time of this run.

variable name	units	description	usage
`year`	integer(4)	year	default = 1
`month`	integer(4)	month	default = 1
`day`	integer(4)	day	default = 1
`hour`	integer(4)	hour	default = 0
`minute`	integer(4)	minute	default = 0
`second`	integer(4)	second	default = 0

Table25.10 Namelist `nml_calendar` that specifies treatment of leap year.

variable name	units	description	usage
`l_force_leap`	logical	Use `l_leap` to decide whether the current year is leap or not.	default = `.false.`, the realistic calendar is followed
`l_leap`	logical	the current year is a leap year or not.	default = `.false.`

Table25.11 Namelist `nml_stdout` that specifies standard output (program log).

variable name	units	description	usage
`l_stdout2file`	logical	Standard output is written to separate files for MPI processes	default = `.false.`
`file_base_stdout`		file name is *name_model*-`file_base_stdout`-stdout.XXXX (XXXX: mpi process number)	
`l_debug`	logical	debug mode	default = `.false.`

25.5 Monitoring

This section summarizes outputs of states in experiments (so called history data) used for monitoring and analyses.

25.5.1 History of the ocean and ice models

Specification of history outputs is common in the ocean and sea ice models. Users prepare a namelist file named NAMELIST.*name_model*.MONITOR dedicated to history outputs (*name_model* can be specified by NAME_MODEL in configure.in, default=OGCM), and write the following namelist blocks, nml_history, in it as needed,

An example namelist for mean temperature output

```
&nml_history
  name          = 'temperature'
  file_base     = 'result/hs_t'
  interval_step = 48
  [suffix       = 'day']
/
```

where name specifies the variable name to be output、file_base the basename of the history files, interval_step the output interval in terms of the integration time step. An option item of suffix specifies the depth of calendar date and time used in the file suffix. In the above example, the time-mean temperature is written to the file, result/hs_t.YYYYMMDD, at every 48-th time step.

History files are saved in a Fortran direct-access format, which can be read by a following program.

Program to read ocean model history data

```
character(8)         :: date = '20010101'  !- for 1JAN2001
real(4)              :: d(imut,jmut,km)
integer(4),parameter :: nu = 10 ! device number

open(nu, file='result/hs_t.'//date, form='unformatted', &
  & access='direct', recl=imut*jmut*km*4)
read(nu,rec=1) d
close(nu)
```

A GrADS control file to visualize data is also made by default (result/hs_t.ctl in the above example).

The state variables that can be monitored by namelist nml_history are listed in docs/README_Monitor.md. There are many options in nml_history, such as netCDF output, snapshot output, averaging in the model region, output in a specified sub region, addition of an offset value, multiplication by a factor, and so on. See docs/README_Monitor.md for the available options.

History outputs of the sea ice model are also specified by nml_history in the same manner as the ocean model. State variables that can be monitored are listed in the sea ice section of docs/README_Monitor.md. The format of output files is also common, except that a missing value for grids of ocean without sea ice should be different from the one for land grids. By default, the value of 0.e0 is used for the former, while -9.99e33 for the latter. (These values can be changed by nml_seaice_hst written in NAMELIST.*name_model*.)

25.6 MRI.COM eXecution Environment (MXE)

We have been developing a package "MRI.COM eXecution Environment (MXE)" that aggregates programs for preprocessing, execution, postprocessing, and analysis of MRI.COM experiments. By using prepared tools, users can relatively easily perform complex work of model building and various analyses. In fact, this package is also used as a common basis for developing various ocean models at the Meteorological Research Institute.

The MXE package includes the following tools.

- Preprocessing tools for creating experiment setup files (directory prep/)
- Directory template for running MRI.COM (exp/)
- Postprocessing tools for visualizing output files (postp/)
- Fortran analysis tools (anl/)
- Python analysis tools (anlpy/)
- A Fortran library that provides basic subroutines such as grid and topography information (lib/)

The main programs are written in Fortran and sample shell scripts are also included for execution. Several tools in the package have been confirmed by unit testing. This package is managed independently from MRI.COM, but it is provided to users as an accompanying material. Since it is often updated like MRI.COM, it is recommended to use the latest version. For details on how to use MXE, see the file `README-MXE.md` in the top directory (in Japanese).

25.7 Appendix

25.7.1 Web site

See the web page https://mri-ocean.github.io/ for getting MRI.COM.

25.7.2 Model options

The model options are listed on Table 25.12. Only major options are listed here. Description about those related to bio-geochemical models can be found in Chapter 19. The description of all options for the latest version can be found in `src/README.Options`. Though an expression like `OGCM_PARALLEL` is used in the source program, `OGCM_` is omitted when users specify options in `configure.in`.

Table25.12 Description of Model Options

Model option	Description
`BBL`	uses the bottom boundary layer model
`BIHARMONIC`	uses biharmonic operator for both horizontal viscosity and diffusion (*) If `ISOPYCNAL` is also selected, the biharmonic form is used only for viscosity and not for diffusion.
`BULKNCAR`	Large and Yeager (2004; 2009) is used for the surface flux bulk formula. This option corresponds to the COREs. (*) `BULKNCAR` is available only for `HFLUX` case.
`CALALBSI`	Sea-ice albedo is calculated using sea-ice conditions according to Los-Alamos model instead of using a constant value
`CALPP`	considers the time variation of pressure for the equation of state
`CARBON`	bio-geochemical process is included (*) `numtrc_p`= 4 for Obata-Kitamura model; `numtrc_p`= 8 for NPZD model
`CBNHSTRUN`	atmospheric pCO_2 is given from file (*) use with `CARBON`
`CHLMA94`	shortwave penetration scheme with chlorophyll concentration by Morel and Antoine (1994) (*) use with `NPZD` and `SOLARANGLE`
`CYCLIC`	uses zonally cyclic condition
`DIFAJS`	sets large vertical diffusion coefficient ($1.0\,m^2\,s^{-1}$) between unstable points instead of convective adjustment
`F2003`	Program uses Fortran 2003 features
`FSVISC`	calculates viscosity explicitly in the barotropic momentum equation
`GMANISOTROP`	Anisotropic horizontal variation of thickness diffusion is used (*) use with `ISOPYCNAL`
`GMTAPER`	Taper GM vector stream function near the sea surface (*) cannot be used with `SLIMIT`, `GMANISOTROP`, `AFC`
`GMVAR`	Horizontal thickness diffusion is allowed to vary in horizontal
`GLS`	Generic length scale model of Umlauf and Burchard (2003)
`HFLUX`	calculates sea surface heat flux using bulk formula
`ICE`	sea ice is included

Continued on next page

Table 25.12 – continued from previous page

Model option	Description
ICECLIM	reading climatological sea-ice fractional area from file
ISOPYCNAL	uses isopycnal diffusion and Gent-McWilliams' parameterization for eddy induced tracer transport velocity (thickness diffusion)
ISOTAPER	Taper isopycnal diffusion coefficient (ISOPYCNAL)
MELYAM	uses Mellor and Yamada Level 2.5 for mixed layer model
MPDATAADVEC	uses MPDATA for tracer advection
NOHKIM	uses Noh's mixed layer model
NPZD	NPZD process is included (*) use with CARBON
OFFNESTPAR	Used as the lower-resolution part of an off-line one-way nesting calculation
OFFNESTSUB	Used as the higher-resolution part of an off-line one-way nesting calculation
PARALLEL	parallel calculation using MPI. The number of zonally and meridionally partitioned regions should be specified as NPARTX and NPARTY, respectively.
PARENT	executed as low resolution model of the on-line nesting calculation
RUNOFF	uses river runoff data (*) available only for WFLUX
SIDYN	sea-ice dynamics model with EVP rheology (*) available only for ICE
SLIMIT	Tapers thickness diffusion near the sea surface
SLP	Sea surface is elevated/depressed according to surface atmospheric pressure
SMAGHD	uses the Smagorinsky viscosity coefficient multiplied by a constant ratio as the horizontal diffusion coefficient (*) available only for SMAGOR
SMAGOR	uses the Smagorinsky parameterization for horizontal viscosity
SOLARANGLE	solar insolation angle is considered in calculating shortwave penetration
SOMADVEC	uses second order moment advection by Prather (1986)
SPHERICAL	calculates scale factor semi-analytically for spherical coordinates
SUB	executed as a high resolution model of the on-line nesting calculation
TAUBULK	calculates the wind stress using bulk formulae by reading wind speed over the ocean
TDEW	reads dew-point temperature and converts to specific humidity (*) available only for HFLUX
TIDE	Tide producing forcing is activated
TRCBIHARM	uses biharmonic operator for horizontal diffusion (*) Should not be used with ISOPYCNAL
TRIPOLAR	Tripolar system is used to construct model grids of a global model (*) Cannot be used with SPHERICAL
UTZQADVEC	"UTOPIA" and "QUICKEST" scheme can be used for horizontal and vertical tracer advection with ultimate limiter
VISANISO	Anisotropic viscosity coefficients are used (*) use with VIS9P
VIS9P	calculates the viscosity using adjacent 9 grid points
VISBIHARM	uses biharmonic operator for both horizontal viscosity
VMBG3D	reads 3-D vertical viscosity and diffusion coefficients from a file
VVDIMP	calculates the vertical diffusion/viscosity by implicit method

Continued on next page

Table 25.12 – continued from previous page

Model option	Description
	(*) it is automatically loaded if any mixed layer model is used or `ISOPYCNAL` option is selected
`WADJ`	adjusts sea surface freshwater flux every time step to keep its global sum to be zero (*) available only for `WFLUX`
`WFLUX`	uses the sea surface freshwater flux to force the model
`MOVE`	used as ocean module for data assimilation (MOVE) system
`SCUP`	use simple coupler (SCUP) library
`SCUPCGCM`	used as an ocean module for a coupled model using scup for communication
`SCUPNEST`	on-line nesting (*) use with `SCUP`