Barotropic–baroclinic time splitting: Requirements and Design

MPAS Development Team

July 26, 2012

Contents

Chapter 1

Summary

Split Barotropic–baroclinic timestepping methods are required in ocean models to increase the timestep length and hence increase computational efficiency. The proposed implementation follows Higdon (2005) and has been implemented and tested in prototype code. The method differs from traditional splitting as the barotropic terms are subcycled explicitly rather than treated implicitly, as in POP.

This document only addresses split exlicit time stepping in z-level coordinates. This will be applied to isopycnal coordinates at a later time. See the ALE vertical coordinate design document.

The split explicit method consists of the following steps: decompose the velocity into barotropic and baroclinic components; take a large timestep with the baroclinic velocities, computing the vertical mean forcing $\overline{\mathbf{G}}$; subcycle the barotropic velocity with small explicit timesteps; add velocities, compute other variables. This process is repeated once more in the Higdon (2005) presentation, but in general may be iterated many times.

Chapter 2

Requirements

2.1 Requirement: A split time-stepping in z-level MPAS

Date last modified: 2011/05/4
Contributors: Mark, Todd

The algorithm should follow Higdon (2005) section 2.3, but with alterations for z-level variables. Input variables should be provided for: timestepping type, number of split explicit iterations, number of barotropic subcycles, and number of baroclinic Coriolis iterations. A unsplit version will be provided that is identical to split explicit but where the full velocity is solved for in the baroclinic stage, and nothing is done in the barotropic stage.

Chapter 3

Algorithmic Formulations

3.1 MPAS-Ocean time splitting, z-level

The MPAS-ocean z-level formulation solves the following equations for thickness, momentum, and tracers at layer k:

$$\frac{\partial h_k}{\partial t} + \nabla \cdot \left(h_k^{edge} \mathbf{u}_k \right) + \frac{\partial}{\partial z} \left(h_k w_k \right) = 0, \tag{3.1}$$

$$\frac{\partial \mathbf{u}_k}{\partial t} + \frac{1}{2} \nabla \left| \mathbf{u}_k \right|^2 + (\mathbf{k} \cdot \nabla \times \mathbf{u}_k) \mathbf{u}_k^\perp + f \mathbf{u}_k^\perp + w_k^{edge} \frac{\partial \mathbf{u}_k}{\partial z} = -\frac{1}{\rho_0} \nabla p_k + \nu_h \nabla^2 \mathbf{u}_k + \frac{\partial}{\partial z} \left(\nu_v \frac{\partial \mathbf{u}_k}{\partial z} \right), \tag{3.2}$$

$$\frac{\partial h_k \varphi_k}{\partial t} + \nabla \cdot \left(h_k^{edge} \varphi_k^{edge} \mathbf{u}_k \right) + \frac{\partial}{\partial z} \left(h_k \varphi_k w_k \right) = \nabla \cdot \left(h_k^{edge} \kappa_h \nabla \varphi_k \right) + h_k \frac{\partial}{\partial z} \left(\kappa_v \frac{\partial \varphi_k}{\partial z} \right). \tag{3.3}$$

The layer thickness h, vertical velocity w, pressure p, and tracer φ, are cell-centered quantities, while the horizontal velocity $\mathbf{u}$ and *edge* superscript are variables located at cell edges. Define the barotopic and baroclinic velocities as

$$\overline{\mathbf{u}} = \sum_{k=1}^{N^{edge}} h_k^{edge} \mathbf{u}_k \Bigg/ \sum_{k=1}^{N^{edge}} h_k^{edge} \tag{3.4}$$

$$\mathbf{u}'_k = \mathbf{u}_k - \overline{\mathbf{u}}, \quad k = 1 \ldots N \tag{3.5}$$

$$\zeta = h_1 - \Delta z_1 \tag{3.6}$$

Here ζ is the sea surface height perturbation and Δz_1 is the top layer thickness with zero perturbation. The barotropic thickness and momentum equations are

$$\frac{\partial \zeta}{\partial t} + \nabla \cdot \left(\overline{\mathbf{u}} \sum_{k=1}^{N^{edge}} h_k^{edge} \right) = 0, \tag{3.7}$$

$$\frac{\partial \overline{\mathbf{u}}}{\partial t} + f \overline{\mathbf{u}}^\perp = -g \nabla \zeta + \overline{\mathbf{G}}, \tag{3.8}$$

where $\overline{\mathbf{G}}$ includes all remaining terms in the barotropic equation. Subtracting the barotropic equation (3.8) from the total momentum equation (3.2), one obtains the baroclinic momentum equation,

$$\frac{\partial \mathbf{u}'_k}{\partial t} + \frac{1}{2} \nabla \left| \mathbf{u}_k \right|^2 + (\mathbf{k} \cdot \nabla \times \mathbf{u}_k) \mathbf{u}_k^\perp + f \mathbf{u}_k'^\perp + w_k \frac{\partial \mathbf{u}_k}{\partial z} \tag{3.9}$$

$$= g \nabla \zeta - \frac{1}{\rho_0} \nabla p_k + \nu_h \nabla^2 \mathbf{u}_k + \frac{\partial}{\partial z} \left(\nu_v \frac{\partial \mathbf{u}_k}{\partial z} \right) - \overline{\mathbf{G}}, \tag{3.10}$$

Consolidating terms for convenience, we can rewrite this as

$$\frac{\partial \mathbf{u}'_k}{\partial t} = -f\mathbf{u}'^{\perp}_k + \mathbf{T}(\mathbf{u}_k, w_k, p_k) + g\nabla\zeta - \overline{\mathbf{G}}, \tag{3.11}$$

$$\mathbf{T}(\mathbf{u}_k, w_k, p_k) = -\frac{1}{2}\nabla\left|\mathbf{u}_k\right|^2 - (\mathbf{k}\cdot\nabla\times\mathbf{u}_k)\mathbf{u}^{\perp}_k - w_k\frac{\partial \mathbf{u}_k}{\partial z} - \frac{1}{\rho_0}\nabla p_k + \nu_h\nabla^2\mathbf{u}_k + \frac{\partial}{\partial z}\left(\nu_v\frac{\partial \mathbf{u}_k}{\partial z}\right). \tag{3.12}$$

For z-level coordinates, set $dh_k/dt = 0$ in the continuity equation (3.1) for $k = 2\ldots N$, and solve for the vertical velocity at layer interfaces with

$$w^{top}_{N+1} = 0, \quad w^{top}_1 = 0 \tag{3.13}$$

$$w^{top}_k = w^{top}_{k+1} - \nabla\cdot(\Delta z_k\mathbf{u}_k), \quad k = N\ldots 2 \tag{3.14}$$

To summarize the equation set,

barotropic momentum
$$\frac{\partial \overline{\mathbf{u}}}{\partial t} = -f\overline{\mathbf{u}}^{\perp} - g\nabla\zeta + \overline{\mathbf{G}}, \tag{3.15}$$

baroclinic momentum
$$\frac{\partial \mathbf{u}'_k}{\partial t} = -f\mathbf{u}'^{\perp}_k + \mathbf{T}(\mathbf{u}_k, w_k, p_k) + g\nabla\zeta - \overline{\mathbf{G}},, \quad k = 1\ldots N, \tag{3.16}$$

total momentum
$$\mathbf{u}_k = \overline{\mathbf{u}} + \mathbf{u}'_k, \ \ k = 1\ldots N, \tag{3.17}$$

barotropic continuity
$$\frac{\partial \zeta}{\partial t} + \nabla\cdot\left(\overline{\mathbf{u}}\sum_{k=1}^{N^{edge}} h^{edge}_k\right) = 0. \tag{3.18}$$

baroclinic continuity
$$w^{top}_k = w^{top}_{k+1} - \nabla\cdot(\Delta z_k\mathbf{u}_k), \quad k = N\ldots 2, \tag{3.19}$$

baroclinic continuity, top
$$\frac{\partial h_1}{\partial t} = w^{top}_2 - \nabla\cdot\left(h^{edge}_1\mathbf{u}_1\right). \tag{3.20}$$

For the split system, sea surface height is overconstrained because both (3.18) and (3.20) provide SSH information. To enforce consistency between the barotropic and baroclinic equation set, replace (3.20) with

$$h_1 = \Delta z_1 + \zeta \tag{3.21}$$

for the split system. For the unsplit algorithm, the barotopric equations (3.15) and (3.18) are not used, so (3.20) is used to find the top layer thickness.

3.2 Split explicit time stepping algorithm, z-level

Prepare variables before first iteration
Always use most recent available for forcing terms. The first time, use end of last timestep.

$$\mathbf{u}_k^* = \mathbf{u}_{k,n}, \quad w_k^* = w_{k,n}, \quad p_k^* = p_{k,n}, \quad \varphi_k^* = \varphi_{k,n}, \tag{3.22}$$

$$h_{k,*} = h_{k,n}, \quad h_{k,*}^{edge} = h_{k,n}^{edge}, \zeta_* = \zeta_n, \quad \mathbf{u}_{k,*}^{bolus} = \mathbf{u}_{k,n}^{bolus} \tag{3.23}$$

$$\overline{\mathbf{u}}_n = \sum_{k=1}^{N^{edge}} h_{k,n}^{edge} \mathbf{u}_{k,n} \Big/ \sum_{k=1}^{N^{edge}} h_{k,n}^{edge}, \text{ on start-up only.} \tag{3.24}$$

$$\text{Otherwise, } \overline{\mathbf{u}}_n \text{ from previous step.} \tag{3.25}$$

$$\mathbf{u}'_{k,n} = \mathbf{u}_{k,n} - \overline{\mathbf{u}}_n \tag{3.26}$$

$$\mathbf{u}'_{k,n+1/2} = \mathbf{u}'_{k,n} \tag{3.27}$$

The full algorithm, Stages 1–3 are iterated. This is typically done using two iterations, like a predictor–corrector timestep. The flag for the number of these large iterations is `config_n_ts_iter`.

Stage 1: Baroclinic velocity (3D), explicit with long timestep
Iterate on linear Coriolis term only.

$$\text{compute } \mathbf{T}^u(\mathbf{u}_k^*, w_k^*, p_k^*) + g\nabla\zeta^* \tag{3.28}$$

$$\text{compute weights, } \omega_k = h_{k,*}^{edge} \Big/ \sum_{k=1}^{N^{edge}} h_{k,*}^{edge} \tag{3.29}$$

$$\left\{\begin{array}{l} \text{compute } \mathbf{u}'^{\perp}_{k,n+1/2} \text{ from } \mathbf{u}'_{k,n+1/2} \\ \tilde{\mathbf{u}}'_{k,n+1} = \mathbf{u}'_{k,n} + \Delta t \left(-f\mathbf{u}'^{\perp}_{k,n+1/2} + \mathbf{T}^u(\mathbf{u}_k^*, w_k^*, p_k^*) + g\nabla\zeta^* \right) \\ \overline{\mathbf{G}} = \frac{1}{\Delta t} \sum_{k=1}^{N^{edge}} \omega_k \tilde{\mathbf{u}}'_{k,n+1} \text{ (unsplit: } \overline{\mathbf{G}} = 0 \text{)} \\ \mathbf{u}'_{k,n+1} = \tilde{\mathbf{u}}'_{k,n+1} - \Delta t \overline{\mathbf{G}} \\ \mathbf{u}'_{k,n+1/2} = \frac{1}{2}\left(\mathbf{u}'_{k,n} + \mathbf{u}'_{k,n+1} \right) \\ \text{boundary update on } \mathbf{u}'_{k,n+1/2} \end{array}\right. \qquad l = 1 \dots L \tag{3.30}$$

$$\tag{3.31}$$

The bracketed computation is iterated L times. The default method is to use $L = 1$ on the first time through Stages 1–3, and $L = 2$ the second time through. This is set by the flags `config_n_bcl_iter_beg = 1, config_n_bcl_iter_mid = 2, config_n_bcl_iter_end = 2`. In the case of two iterations through Stages 1–3, the flag `config_n_bcl_iter_mid` is not used.

Stage 2: Barotropic velocity (2D), explicitly subcycled
Advance $\overline{\mathbf{u}}$ and ζ as a coupled system through $2J$ subcycles, ending at time $t + 2\Delta t$.

velocity predictor step:

$$\text{compute } \overline{\mathbf{u}}^{\perp}_{n+(j-1)/J} \text{ from } \overline{\mathbf{u}}_{n+(j-1)/J} \tag{3.32}$$

$$\tilde{\overline{\mathbf{u}}}_{n+j/J} = \overline{\mathbf{u}}_{n+(j-1)/J} + \frac{\Delta t}{J}\left(-f\overline{\mathbf{u}}^{\perp}_{n+(j-1)/J} - g\nabla\zeta_{n+(j-1)/J} + \overline{\mathbf{G}}_j\right) \tag{3.33}$$

$$\text{boundary update on } \tilde{\overline{\mathbf{u}}}_{n+j/J} \tag{3.34}$$

SSH predictor step:

$$\zeta^{edge}_{n+(j-1)/J} = Interp(\zeta_{n+(j-1)/J}) \tag{3.35}$$

$$\tilde{\mathbf{F}}_j = \left((1-\gamma_1)\overline{\mathbf{u}}_{n+(j-1)/J} + \gamma_1\tilde{\overline{\mathbf{u}}}_{n+j/J}\right)\left(\zeta^{edge}_{n+(j-1)/J} + H^{edge}\right) \tag{3.36}$$

$$\tilde{\zeta}_{n+j/J} = \zeta_{n+(j-1)/J} + \frac{\Delta t}{J}\left(-\nabla\cdot\tilde{\mathbf{F}}_j\right) \tag{3.37}$$

$$\text{boundary update on } \tilde{\zeta}_{n+j/J} \tag{3.38}$$

velocity corrector step:

$$\text{compute } \tilde{\overline{\mathbf{u}}}^{\perp}_{n+j/J} \text{ from } \tilde{\overline{\mathbf{u}}}_{n+j/J} \tag{3.39}$$

$$\overline{\mathbf{u}}_{n+j/J} = \overline{\mathbf{u}}_{n+(j-1)/J} + \frac{\Delta t}{J}\left(-f\tilde{\overline{\mathbf{u}}}^{\perp}_{n+j/J} - g\nabla\left((1-\gamma_2)\zeta_{n+(j-1)/J} + \gamma_2\tilde{\zeta}_{n+j/J}\right) + \overline{\mathbf{G}}_j\right) \tag{3.40}$$

$$\text{boundary update on } \overline{\mathbf{u}}_{n+j/J} \tag{3.41}$$

SSH corrector step:

$$\tilde{\zeta}^{\,edge}_{n+j/J} = Interp\left((1-\gamma_2)\zeta_{n+(j-1)/J} + \gamma_2\tilde{\zeta}_{n+j/J}\right) \tag{3.42}$$

$$\mathbf{F}_j = \left((1-\gamma_3)\overline{\mathbf{u}}_{n+(j-1)/J} + \gamma_3\overline{\mathbf{u}}_{n+j/J}\right)\left(\tilde{\zeta}^{\,edge}_{n+j/J} + H^{edge}\right) \tag{3.43}$$

$$\zeta_{n+j/J} = \zeta_{n+(j-1)/J} + \frac{\Delta t}{J}\left(-\nabla\cdot\mathbf{F}_j\right) \tag{3.44}$$

$$\text{boundary update on } \zeta_{n+j/J} \tag{3.45}$$

Repeat $j = 1 \ldots 2J$ to step through the barotropic subcycles. There are two predictor and two corrector steps for each subcycle. At the end of $2J$ subcycles, we have progressed through two baroclinic timesteps, i.e. through $2\Delta t$. In the code, `config_n_btr_subcycles` is J, while `config_btr_subcycle_loop_factor=2` is the "2" coefficient in $2J$.

The input flag `config_btr_solve_SSH2=.true.` runs the algorithm as shown, while `.false.` does not include the SSH corrector step. Here H^{edge} is the total column depth without SSH perturbations, that is, from the higher cell adjoining an edge to $z = 0$.

The velocity corrector step may be iterated to update the velocity in the Coriolis term. The number of iterations is controlled by `config_n_btr_cor_iter`, and is usually set to two.

Stage 2 continued

The coefficients $(\gamma_1, \gamma_2, \gamma_3)$ control weighting between the old and new variables in the predictor velocity, corrector SSH gradient, and corrector velocity, respectively. These are typically set as $(\gamma_1, \gamma_2, \gamma_3) = (0.5, 1, 1)$, but this is open to investigation. These flags are `config_btr_gam1_uWt1, config_btr_gam2_SSHWt1, config_btr_gam3_uWt2`.

The baroclinic forcing $\overline{\mathbf{G}}_j$ may vary over the barotropic subcycles, as long as $\frac{1}{2J}\sum_{j=1}^{2J} \overline{\mathbf{G}}_j = \overline{\mathbf{G}}$. This option is not currently implemented in the code.

$$\overline{\mathbf{u}}_{avg} = \frac{1}{2J+1}\sum_{j=0}^{2J} \overline{\mathbf{u}}_{n+j/J} \tag{3.46}$$

$$\overline{\mathbf{F}} = \frac{1}{2J}\sum_{j=1}^{2J} \mathbf{F}_j \tag{3.47}$$

$$\text{boundary update on } \overline{\mathbf{F}} \tag{3.48}$$

$$\mathbf{u}^{corr} = \left(\overline{\mathbf{F}} - \sum_{k=1}^{N^{edge}} h_{k,*}^{edge} \left(\overline{\mathbf{u}}_{avg} + \mathbf{u}'_{k,n+1/2} + \mathbf{u}_{k,*}^{bolus} \right) \right) \Bigg/ \sum_{k=1}^{N^{edge}} h_{k,*}^{edge} \tag{3.49}$$

$$\mathbf{u}_k^{tr} = \overline{\mathbf{u}}_{avg} + \mathbf{u}'_{k,n+1/2} + \mathbf{u}_{k,*}^{bolus} + \mathbf{u}^{corr} \tag{3.50}$$

where $\mathbf{u}_{k,*}^{bolus}$ is the GM bolus velocity computed at the end of stage 3, and $\mathbf{u}^{tr}$ is the transport velocity used in the advection terms for for thickness and tracers.

For unsplit explicit, skip all computations in stage 2. Instead, set:

$$\overline{\mathbf{u}}_{avg} = 0 \tag{3.51}$$

$$\mathbf{u}_k^{tr} = \mathbf{u}'_{k,n+1/2} + \mathbf{u}_{k,*}^{bolus} \tag{3.52}$$

Stage 3: update thickness, tracers, density and pressure

$$w_k^{*\,top} = w_{k+1}^{*\,top} - \nabla \cdot \left(\Delta z_k \mathbf{u}_k^{tr}\right), \quad k = N \dots 2. \tag{3.53}$$

$$T_k^h = \left(-\nabla \cdot \left(h_k^{*\,edge} \mathbf{u}_k^{tr}\right) - \frac{\partial}{\partial z}\left(h_k^* w_k^*\right)\right) \tag{3.54}$$

$$T^\varphi = \left(-\nabla \cdot \left(h_k^{*\,edge} \varphi_k^* \mathbf{u}_k^{tr}\right) - \frac{\partial}{\partial z}\left(h_k^* \varphi_k^* w_k^*\right) + \nabla \cdot \left(h_k^* \kappa_h \nabla \varphi_k^*\right) + h_k^* \frac{\partial}{\partial z}\left(\kappa_v \frac{\partial \varphi_k^*}{\partial z}\right)\right) \tag{3.55}$$

$$\text{boundary update on tendencies: } T^h, T^\varphi \tag{3.56}$$

$$h_{k,n+1} = h_{k,n} + \Delta t T_k^h \tag{3.57}$$

$$\varphi_{k,n+1} = \frac{1}{h_{k,n+1}} \left[h_{k,n} \varphi_{k,n} + \Delta t T_k^\varphi\right] \tag{3.58}$$

Reset variables

$$\begin{array}{ll}
\textbf{if iterating} & \textbf{after final iteration} \\
\mathbf{u}'_{k,*} = \mathbf{u}'_{k,n+1/2} & \mathbf{u}'_{k,n+1} \text{ from stage 1} \\
\overline{\mathbf{u}}_* = \overline{\mathbf{u}}_{avg} \text{ from stage 2} & \overline{\mathbf{u}}_{n+1} = \overline{\mathbf{u}}_{avg} \text{ from stage 2} \\
\mathbf{u}_{k,*} = \overline{\mathbf{u}}_* + \mathbf{u}'_{k,*} & \mathbf{u}_{k,n+1} = \overline{\mathbf{u}}_{n+1} + \mathbf{u}'_{k,n+1} \\
h_{k,*} = \frac{1}{2}\left(h_{k,n} + h_{k,n+1}\right) & h_{k,n+1} \text{ from stage 3} \\
\varphi_{k,*} = \frac{1}{2}\left(\varphi_{k,n} + \varphi_{k,n+1}\right) & \varphi_{k,n+1} \text{ from stage 3} \\
\text{diagnostics:} & \text{diagnostics:} \\
\rho_k^* = EOS(T_k^*, S_k^*) & \rho_{k,n+1} = EOS(T_{k,n+1}, S_{k,n+1}) \\
p_k^* = g \sum_{k'=1}^{k-1} \rho_{k'}^* h_{k'} + \frac{1}{2} g \rho_k^* h_k & p_{k,n+1} = g \sum_{k'=1}^{k-1} \rho_{k',n+1} h_{k',n+1} + \frac{1}{2} g \rho_{k,n+1}^* h_{k,n+1} \\
h_{k,*}^{edge} = interp(h_{k,*}) & h_{k,n+1}^{edge} = interp(h_{k,n+1}) \\
\zeta_* = \sum_{k=1}^{kmax} h_{k,*} - H & \zeta_{n+1} = \sum_{k=1}^{kmax} h_{k,n+1} - H \\
\text{compute } \mathbf{u}_{k,*}^{bolus} & \text{compute } \mathbf{u}_{k,n+1}^{bolus}
\end{array} \tag{3.59}$$

where H is the total column height with zero sea surface height.

notes:

1. May be able to not compute phi, rho, p, until the end, i.e. compute (3.58-3.59) last time only.

An explanation of the stage 3 transport velocity $\mathbf{u}_k^{tr}$ used in the tracer equation is as follows. For tracer conservation, (3.58) with constant φ must reduce to the thickness equation for all k. Ignoring diffusion terms, this gives

$$h_k^* = h_{k,n} + \Delta t \left(-\nabla \cdot \left(h_k^{*\,edge} \mathbf{u}_k^{tr} \right) - \left(w_k^{*\,top} - w_{k+1}^{*\,top} \right) \right). \tag{3.60}$$

For $k > 1$, h_k is constant throughout, and one may solve for $w_{k+1}^{*\,top}$, and this leads to equation (3.53). For $k = 1$, we have

$$\begin{aligned} h_1^* &= h_{1,n} + \Delta t \left(-\nabla \cdot \left(h_1^{*\,edge} \mathbf{u}_1^{tr} \right) + w_2^{*\,top} \right) && (3.61) \\ &= h_{1,n} + \Delta t \left(-\nabla \cdot \left(\sum_{k=1}^{N} h_k^{*\,edge} \mathbf{u}_k^{tr} \right) \right) && (3.62) \end{aligned}$$

For consistency between the barotropic and summed baroclinic thickness flux, we must enforce

$$\overline{\mathbf{F}} = \sum_{k=1}^{N} h_k^{*\,edge} \mathbf{u}_k^{tr}. \tag{3.63}$$

To do that, introduce a velocity correction u^{corr} that is vertically constant, so that $\mathbf{u}_k^{tr} = u_k^* + u^{corr}$. Substitute into (3.63) and solve for the velocity correction,

$$u^{corr} = \left(\overline{\mathbf{F}} - \sum_{k=1}^{N^{edge}} h_{k,n}^{*\,edge} u_k^* \right) \Bigg/ \sum_{k=1}^{N^{edge}} h_{k,n}^{*\,edge} \tag{3.64}$$

3.3 Unsplit algorithm, z-level

Prep variables before first iteration
Always use most recent available for forcing terms. The first time, use end of last timestep.

$$\mathbf{u}_k^* = \mathbf{u}_{k,n}, \quad w_k^* = w_{k,n}, \quad p_k^* = p_{k,n}, \quad h_k^{*\,edge} = h_{k,n}^{edge}, \quad \varphi_k^* = \varphi_{k,n} \tag{3.65}$$

$$\overline{\mathbf{u}}_n = 0, \quad \mathbf{u}'_{k,n} = \mathbf{u}_{k,n} - \overline{\mathbf{u}}_n, \quad \mathbf{u}'_{k,n+1/2} = \mathbf{u}'_{k,n} \tag{3.66}$$

Stage 1: Baroclinic velocity (3D) prediction, explicit with long timestep

$$\text{compute } \mathbf{T}(\mathbf{u}_k^*, w_k^*, p_k^*) \tag{3.67}$$

$$\left\{\begin{array}{l} \text{compute } \mathbf{u}'^{\perp}_{k,n+1/2} \text{ from } \mathbf{u}'_{k,n+1/2} \\ \tilde{\mathbf{u}}'_{k,n+1} = \mathbf{u}'_{k,n} + \Delta t \left(-f \mathbf{u}'^{\perp}_{k,n+1/2} + \mathbf{T}(\mathbf{u}_k^*, w_k^*, p_k^*) \right) \\ \overline{\mathbf{G}} = 0 \\ \mathbf{u}'_{k,n+1} = \tilde{\mathbf{u}}'_{k,n+1} - \Delta t \overline{\mathbf{G}} \\ \mathbf{u}'_{k,n+1/2} = \frac{1}{2}\left(\mathbf{u}'_{k,n} + \mathbf{u}'_{k,n+1} \right) \\ \text{boundary update on } \mathbf{u}'_{k,n+1/2} \end{array}\right. \qquad l = 1 \dots L \tag{3.68}$$

$$\mathbf{u}'^*_k = \mathbf{u}'_{k,n+1/2} \tag{3.69}$$

Stage 2: Barotropic velocity (2D) prediction, explicitly subcycled
$\overline{\mathbf{u}}^* = 0$, $\mathbf{u}_k * = \mathbf{u}'^*_k$, no other computations here.

Stage 3: Tracer, density, pressure, vertical velocity prediction
Note that the new $h_1^{*\,edge}$ is computed after ϕ_k^*, so that (3.71) and (3.72) use the same version of $h_1^{*\,edge}$. This ensures that the tracer equation reduces to the thickness equation at layer 1 with constant tracers.

$$w_k^{*\,top} = w_{k+1}^{*\,top} - \nabla \cdot \left(\Delta z_k \mathbf{u}_k^* \right), \quad k = N \dots 2. \tag{3.70}$$

$$h_{1,n+1} = h_{1,n} + \Delta t \left(w_2^{*\,top} - \nabla \cdot \left(h_1^{*\,edge} \mathbf{u}_1^* \right) \right) \tag{3.71}$$

$$\begin{aligned} \varphi_{k,n+1} = \frac{1}{h_{k,n+1}} \Bigg[& h_{k,n}\varphi_{k,n} + \Delta t \Bigg(-\nabla \cdot \left(h_k^{*\,edge} \varphi_k^* \mathbf{u}_k^* \right) - \frac{\partial}{\partial z}\left(h_k^* \varphi_k^* w_k^* \right) \\ & + \nabla \cdot \left(h_k^* \kappa_h \nabla \varphi_k^* \right) + h_k^* \frac{\partial}{\partial z}\left(\kappa_v \frac{\partial \varphi_k^*}{\partial z} \right) \Bigg) \Bigg] \end{aligned} \tag{3.72}$$

$$\text{boundary update on } \varphi_{k,n+1}, h_1^* \tag{3.73}$$

$$\varphi_k^* = \tfrac{1}{2}\left(\varphi_{k,n} + \varphi_{k,n+1} \right), \ \text{(not on last iteration)} \tag{3.74}$$

$$h_1^* = \tfrac{1}{2}\left(h_{1,n} + h_{1,n+1} \right) \ \text{(not on last iteration)} \tag{3.75}$$

$$\mathbf{u}_k^* = \overline{\mathbf{u}}^* + \mathbf{u}'^*_k \tag{3.76}$$

$$h_1^{*\,edge} = Interp(h_1^*) \tag{3.77}$$

$$\rho_k^* = EOS(T_k^*, S_k^*) \tag{3.78}$$

$$p_k^* = g\rho_1^* \left(h_1^* - \frac{1}{2}\Delta z_1 \right) + \frac{g}{2} \sum_{l=2}^{k} \left(\rho_{l-1}^* \Delta z_{l-1} + \rho_l^* \Delta z_l \right) \tag{3.79}$$

Iteration
If iterating, return to stage 1.
If complete, we have $\mathbf{u}_{k,n+1} = \mathbf{u}'_{k,n+1}$ from (3.68), $\varphi_{k,n+1}$ from (3.72), $h_{1,n+1}$ from (3.71). Then compute the full end-of-step diagnostics, including $w_{k,n+1}^{top}$, $\rho_{k,n+1}$ and $p_{k,n+1}$.

3.4 Runge-Kutta Fourth Order algorithm, z-level

Prep variables before first stage

$$\mathbf{u}_{k,n+1} = \mathbf{u}_{k,n}, \;\; h_{k,n+1} = h_{k,n}, \;\; \varphi_{k,n+1} = \varphi_{k,n} h_{k,n} \tag{3.80}$$

$$\mathbf{u}^*_k = \mathbf{u}_{k,n}, \;\; h^*_k = h_{k,n}, \;\; \varphi^*_k = \varphi_{k,n}, \;\; p^*_k = p_{k,n}, \;\; w^*_k = w_{k,n}, \text{ etc.} \tag{3.81}$$

$$a = \left(\frac{1}{2}, \frac{1}{2}, 1, 0\right), \;\; b = \left(\frac{1}{6}, \frac{1}{3}, \frac{1}{3}, \frac{1}{6}\right), \tag{3.82}$$

Iteration

$$\text{do j=1,4} \tag{3.83}$$

$$\text{compute } \mathbf{T}^u(\mathbf{u}^*_k, w^*_k, p^*_k), \;\; \mathbf{T}^h(\mathbf{u}^*_k, w^*_k), \;\; \mathbf{T}^\varphi(\mathbf{u}^*_k, w^*_k, \varphi^*_k) \tag{3.84}$$

$$\text{boundary update on all tendencies: } \mathbf{T}^u, \mathbf{T}^h, \mathbf{T}^\varphi \tag{3.85}$$

$$\mathbf{u}^*_k = \mathbf{u}_{k,n} + a_j \Delta t \mathbf{T}^u_k \tag{3.86}$$

$$h^*_k = h_{k,n} + a_j \Delta t \mathbf{T}^h_k \tag{3.87}$$

$$\varphi^*_k = \frac{1}{h^*_k}\left[h_{k,n}\varphi_{k,n} + a_j \Delta t \mathbf{T}^\varphi_k\right] \tag{3.88}$$

$$\text{compute diagnostics based on } \mathbf{u}^*_k, h^*_k, \varphi^*_k \tag{3.89}$$

$$\mathbf{u}_{k,n+1} = \mathbf{u}_{k,n+1} + b_j \Delta t \mathbf{T}^u_k \tag{3.90}$$

$$h_{k,n+1} = h_{k,n+1} + b_j \Delta t \mathbf{T}^h_k \tag{3.91}$$

$$\varphi_{k,n+1} = \varphi_{k,n+1} + b_j \Delta t \mathbf{T}^\varphi_k \tag{3.92}$$

$$\text{enddo} \tag{3.93}$$

End of step

$$\varphi_{k,n+1} = \varphi_{k,n+1}/h_{k,n+1} \tag{3.94}$$

$$\text{revise } \mathbf{u}_{k,n+1}, \;\; \varphi_{k,n+1} \text{ with implicit vertical mixing} \tag{3.95}$$

$$\text{compute diagnostics based on } \mathbf{u}_{k,n+1}, h_{k,n+1}, \varphi_{k,n+1} \tag{3.96}$$

Chapter 4

Design and Implementation

4.1 Design Solution: split explicit time-stepping in z-level mpas

Date last modified: 2011/05/4
Contributors: Mark, Todd

4.2 New variables

```
namelist character timestep        config_time_integration 'RK4',
 also: 'unsplit','split_explicit'
namelist integer   split_explicit_ts config_n_ts_iter     2
namelist integer   split_explicit_ts config_n_bcl_iter_beg   4
namelist integer   split_explicit_ts config_n_bcl_iter_mid   4
namelist integer   split_explicit_ts config_n_bcl_iter_end   4
namelist integer   split_explicit_ts config_n_btr_subcycles  10
namelist logical   split_explicit_ts config_compute_tr_midstage true

var persistent real   uBtr ( nEdges Time )         2 o  state
var persistent real   ssh ( nCells Time )          2 o  state
var persistent real   uBtrSubcycle ( nEdges Time ) 2 o  state
var persistent real   sshSubcycle ( nCells Time )  2 o  state
var persistent real   FBtr ( nEdges Time )         1 o  state
var persistent real   GBtrForcing ( nEdges Time )  1 o  state
var persistent real   uBcl ( nVertLevelsP1 nEdges Time )  2 o  state
```

4.3 Higdon time splitting code, z-level

Prepare variables before first iteration

`uOld => block % state % time_levs(1) % state % u % array(:,:)` etc.
`uNew => block % state % time_levs(2) % state % u % array(:,:)` etc.
`uBclOld => block % state % time_levs(1) % state % uBcl % array(:,:)` etc.
`uBclNew => block % state % time_levs(2) % state % uBcl % array(:,:)` etc.
`tend_u => block % tend % u % array(:,:)`

$\mathbf{u}_k^* = \mathbf{u}_{k,n}, \;\; w_k^* = w_{k,n}, \;\; p_k^* = p_{k,n}, \;\; \varphi_k^* = \varphi_{k,n}$

`Do nothing.  * variables are already in time_levs(2) slots.`

$\overline{\mathbf{u}}_n = \sum_{k=1}^{N^{edge}} h_{k,n}^{edge} \mathbf{u}_{k,n} \Big/ \sum_{k=1}^{N^{edge}} h_{k,n}^{edge}$

`uBtrOld(iEdge) =`

$\mathbf{u}'_{k,n} = \mathbf{u}_{k,n} - \overline{\mathbf{u}}_n$

`uBclOld = uOld - uBtrOld`

$\mathbf{u}'_{k,n+1} = \mathbf{u}'_{k,n}$

`uBclNew = uBclOld`

Stage 1: Baroclinic velocity (3D) prediction, explicit with long timestep

compute $\mathbf{T}(\mathbf{u}_k^*, w_k^*, p_k^*) + g\nabla\zeta^*$

```
put in tend_u, separate out linear Coriolis term
do j=1,config_n_bcl_iter
```

  compute $\mathbf{u}'^{\perp}_{k,n+1}$ from $\mathbf{u}'_{k,n+1}$

```
  call compute_uPerp(uBclNew,uBclPerp)
  need subroutine and variable just for perp, say uBclPerp - may already be there.
  do iEdge=1,nEdges
    do k=1,maxLevelEdgeTop(iEdge)
```

      $\mathbf{G}_k = -f\mathbf{u}'^{\perp}_{k,n+1} + \mathbf{T}(\mathbf{u}_k^*, w_k^*, p_k^*) + g\nabla\zeta^*$

```
      G(k) = -fEdge(iEdge)*uBclPerp(k,iEdge) + tend_u(k,iEdge)
    enddo
```

    $\overline{\mathbf{G}} = \sum_{k=1}^{N^{edge}} h_{k,n}^{edge} \mathbf{G}_k \Big/ \sum_{k=1}^{N^{edge}} h_{k,n}^{edge}$

```
    GBtrForcing(iEdge) = sum(hOld(k,iEdge)* G(k))/(hOld(1,iEdge) + h2toNZLevel)
    do k=1,maxLevelEdgeTop(iEdge)
```

      $\mathbf{u}'_{k,n+1} = \mathbf{u}'_{k,n} + \Delta t \left(\mathbf{G}_k - \overline{\mathbf{G}}\right)$

```
      uBclNew(k,iEdge) = uBclOld(k,iEdge) + dt*(G(k)-GBtrForcing(iEdge)
    enddo
  enddo
  boundary update on uBclNew
enddo
```

$\mathbf{u}_k'^{*} = \frac{1}{2}\left(\mathbf{u}'_{k,n} + \mathbf{u}'_{k,n+1}\right)$

This is done in stage 2 so we don't overwrite uBclNew

Stage 2: Barotropic velocity (2D) prediction, explicitly subcycled

```
sshSubcycleOld = sshOld
uBtrSubcycleOld = uBtrOld
uBtrNew = aBtrSumCoef(0)*uBtrOld
sshNew = aBtrSumCoef(0)*sshOld
FBtr = 0
do j=1,2*config_n_btr_subcycles
```

$\zeta_{n+j/J}^{edge} = Interp(\zeta_{n+j/J})$

$\mathbf{F}_j = \overline{\mathbf{u}}_{n+j/J} \left(\zeta_{n+j/J}^{edge} + \sum_{k=1}^{N^{edge}} \Delta z_k \right)$

$\zeta_{n+(j+1)/J} = \zeta_{n+j/J} + \frac{\Delta t}{J} \left(-\nabla \cdot \mathbf{F}_j \right)$

```
  do iEdge=1,nEdges
    cell1 = cellsOnEdge(1,iEdge)
    cell2 = cellsOnEdge(2,iEdge)
    flux = u(k,iEdge) * dvEdge(iEdge) * h_edge(k,iEdge)
    FBtr(iEdge) = FBtr(iEdge) + bBtrSumCoef(j)*flux
    tend_ssh(cell1) = tend_ssh(cell1) - flux
    tend_ssh(cell2) = tend_ssh(cell2) + flux
  end do
  do iCell=1,nCells
    sshSubcycleNew(iCell) = sshSubcycleOld(iCell) + tend_ssh(iCell)/areaCell(iCell)
    sshNew(iCell) = sshNew(iCell) + aBtrSumCoef(j)*sshSubcycleNew(iCell)
  end do
```

boundary update on $\zeta_{n+(j+1)/J}$

compute $\overline{\mathbf{u}}_{n+j/J}^{\perp}$ from $\overline{\mathbf{u}}_{n+j/J}$

$\overline{\mathbf{u}}_{n+(j+1)/J} = \overline{\mathbf{u}}_{n+j/J} + \frac{\Delta t}{J} \left(-f \overline{\mathbf{u}}_{n+j/J}^{\perp} - g \nabla \zeta_{n+(j+1)/J} + \overline{\mathbf{G}} \right),$

```
  do iEdge=1,nEdges
    cell1 = cellsOnEdge(1,iEdge)
    cell2 = cellsOnEdge(2,iEdge)
    grad_ssh = - gravity*rho0Inv*( sshSubcycleNew(cell2) &
               - sshSubcycleNew(cell1) )/dcEdge(iEdge)
    uBtrSubCycleNew(iEdge) = uBtrSubcycleOld(iEdge) + dt*(fEdge(iEdge)*uBtrPerp(iEdge)
        - grad_ssh + GBtrForcing(iEdge))
    uBtrNew(iEdge) = uBtrNew(iEdge) + aBtrSumCoef(j)*uBtrSubcycleNew(iEdge)
  end do
```

boundary update on $\overline{\mathbf{u}}_{n+(j+1)/J}$

```
enddo
```

Note: Normalize so that $\sum_{j=0}^{2J} a_j = 1$ and $\sum_{j=1}^{2J} b_j = 1$. Then the following three lines are already done, so that

$\zeta^* = \sum_{j=0}^{2J} a_j \zeta_{n+j/J} \Big/ \sum_{j=0}^{2J} a_j$ is `sshNew`

$\overline{\mathbf{u}}^* = \sum_{j=0}^{2J} a_j \overline{\mathbf{u}}_{n+j/J} \Big/ \sum_{j=0}^{2J} a_j$ is `uBtrNew`

$\overline{\mathbf{F}} = \sum_{j=1}^{2J} b_j \mathbf{F}_j \Big/ \sum_{j=1}^{2J} b_j$ is `FBtr`

$\mathbf{u}_k^* = \overline{\mathbf{u}}^* + \mathbf{u}_k^{\prime *}$

```
uNew(iEdge) = uBtrNew(iEdge) + 0.5*(uBclOld(iEdge) + uBclNew(iEdge))
```

$h_1^* = \Delta z_1 + \zeta^*$

```
hNew(iCell) = hZlevel(1) + sshNew(iCell)
```

boundary update on $\overline{\mathbf{F}}$

Stage 3: Tracer, density, pressure, vertical velocity prediction

$w_k^{*\,top} = w_{k+1}^{*\,top} - \nabla \cdot \left(\Delta z_k \mathbf{u}_k^*\right), \quad k = N \ldots 2.$

$h_1^{*\,edge} = \frac{1}{\mathbf{u}_1^* + \epsilon} \left(\overline{\mathbf{F}} - \sum_{k=2}^{N} h_k u_k^* \right)$

make sure $h_1^{*\,edge}$ is bounded by neighboring cells.

$\varphi_k^* = \frac{1}{h_k^*} \left[h_{k,n}\varphi_{k,n} + \Delta t \left(-\nabla \cdot \left(h_k^{*\,edge} \varphi_k^* \mathbf{u}_k^* \right) - \frac{\partial}{\partial z} \left(h_k^* \varphi_k^* w_k^* \right) + \nabla \cdot \left(h_k^* \kappa_h \nabla \varphi_k^* \right) + h_k^* \frac{\partial}{\partial z} \left(\kappa_v \frac{\partial \varphi_k^*}{\partial z} \right) \right) \right]$

`tracerNew =`

$\rho_k^* = EOS(T_k^*, S_k^*)$

$p_k^* = g\rho_1^* \left(h_1^* - \frac{1}{2}\Delta z_1 \right) + \frac{g}{2} \sum_{l=2}^{k} \left(\rho_{l-1}^* \Delta z_{l-1} + \rho_l^* \Delta z_l \right)$

Iteration

If iterating, return to stage 1.

If complete, then:

$\mathbf{u}_k^* = \overline{\mathbf{u}}^* + \mathbf{u}_{k,n+1}^{\prime *}$

```
uNew(iEdge) = uBtrNew(iEdge) + uBclNew(iEdge))
```

$w_k^{*\,top} = w_{k+1}^{*\,top} - \nabla \cdot \left(\Delta z_k \mathbf{u}_k^*\right), \quad k = N \ldots 2.$

$h_{1,n+1} = h_1^* \;\; \varphi_{k,n+1} = \varphi_k^*, \;\; \rho_{k,n+1} = \rho_k^*, \;\; p_{k,n+1} = p_k^*$

Do nothing, starred variables are already `hNew, tracerNew, rhoNew, pNew` etc.

Compute the full end-of-step diagnostics.

```
call compute_solve_diagnostics
```

4.4 Runge-Kutta Fourth Order code, z-level

Prep variables before first stage

`uOld => block % state % time_levs(1) % state % u % array(:,:)` etc.
`uNew => block % state % time_levs(2) % state % u % array(:,:)` etc.
`uProvis => provis % u % array(:,:)` etc.
`tend_u => block % tend % u % array(:,:)`

$\mathbf{u}_{k,n+1} = \mathbf{u}_{k,n}, \;\; h_{k,n+1} = h_{k,n}, \;\; \varphi_{k,n+1} = \varphi_{k,n} h_{k,n}$

```
uNew = uOld
hNew = hOld
tracerNew = tracerOld
```

$\mathbf{u}^*_k = \mathbf{u}_{k,n}, \;\; h^*_k = h_{k,n}, \;\; \varphi^*_k = \varphi_{k,n}, \;\; p^*_k = p_{k,n}, \;\; w^*_k = w_{k,n},$ etc.

`call allocate_state(provis, &` etc.
`call copy_state(provis, block % state % time_levs(1) % state)`

$a = \left(\frac{1}{2}, \frac{1}{2}, 1, 0\right), \;\; b = \left(\frac{1}{6}, \frac{1}{3}, \frac{1}{3}, \frac{1}{6}\right)$

`rk_substep_weights(1) = dt/2.`, etc. `rk_weights(1) = dt/6.`, etc.

Iteration

`do rk_step=1,4`

compute $\mathbf{T}^u(\mathbf{u}^*_k, w^*_k, p^*_k), \;\; \mathbf{T}^h(\mathbf{u}^*_k, w^*_k), \;\; \mathbf{T}^\varphi(\mathbf{u}^*_k, w^*_k, \varphi^*_k)$

```
call compute_tend(block % tend, provis, block % diagnostics, block % mesh)
call compute_scalar_tend(block % tend, provis, block % diagnostics, block % mesh)
```

$\mathbf{u}^*_k = \mathbf{u}_{k,n} + a_j \Delta t \mathbf{T}^u_k$

`uProvis = uOld + rk_substep_weights(rk_step) * tend_u`

$h^*_k = h_{k,n} + a_j \Delta t \mathbf{T}^h_k$

`hProvis = hOld + rk_substep_weights(rk_step) * tend_h`

$\varphi^*_k = \frac{1}{h^*_k}\left[h_{k,n}\varphi_{k,n} + a_j \Delta t \mathbf{T}^\varphi_k\right]$

`tracerProvis = (hOld*tracerOld + rk_substep_weights(rk_step) * tracer_tend)/hProvis`

compute diagnostics based on $\mathbf{u}^*_k, h^*_k, \varphi^*_k$

`call compute_solve_diagnostics(dt, provis, block % mesh)`

$\mathbf{u}_{k,n+1} = \mathbf{u}_{k,n+1} + b_j \Delta t \mathbf{T}^u_k$

`uNew = uNew + rk_weights(rk_step) * tend_u`

$h_{k,n+1} = h_{k,n+1} + b_j \Delta t \mathbf{T}^h_k$

`hNew = hNew + rk_weights(rk_step) * tend_h`

$\varphi_{k,n+1} = \varphi_{k,n+1} + b_j \Delta t \mathbf{T}^\varphi_k$

`tracerNew = tracerNew + rk_weights(rk_step) * tracer_tend`

`enddo`

End of step

$\varphi_{k,n+1} = \varphi_{k,n+1}/h_{k,n+1}$

`tracers(:,k,iCell) = tracers(:,k,iCell) / h(k,iCell)`

revise $\mathbf{u}_{k,n+1}, \;\; \varphi_{k,n+1}$ with implicit vertical mixing

`call compute_vertical_mix_coefficients(block % state % time_levs(2) % state, block % diagnos`
`call tridiagonal_solve(A,C,A,u(:,iEdge),uTemp,maxLevelEdgeTop(iEdge))` etc.

compute diagnostics based on $\mathbf{u}_{k,n+1}, h_{k,n+1}, \varphi_{k,n+1}$

`call compute_solve_diagnostics(dt, block % state % time_levs(2) % state, block % mesh)`

Chapter 5

Testing

5.1 Testing and Validation: split explicit time-stepping in z-level mpas

Date last modified: 2011/05/04
Contributors: (add your name to this list if it does not appear)

Testing: Compare Runge-Kutta versus unsplit and split explicit.