**A Wetting and Drying (W/D) Method for NEMO-shelf**

**Based on Flux Limiter**

Hedong Liu and Jason Holt (NOC, UK)

NEMO uses structured C-grid, hence W/D needs to deal with at least two types of grid cells: velocity cells and tracer cells. The essences of current method are to keep the positivity of water mass and avoid overshooting of velocity on the velocity cells. At the same time, this method must also respect the properties of local and global conservation. There are quite a lot of existing methods for W/D, (see Balzano, A, 1998 and Medeiros, S. C. and Hagen, S. C, 2013). But there are still some issues, such as the restriction of CFL condition, properties of conservation. In the current method, a flux limiter has been developed to take care of the mass positivity while a new barotropic pressure gradient coefficient is developed to prevent the velocity overshooting at the wetting or drying cells on a steep slope.

In the following parts, the design of the flux limiter and its applications on the continuity equation, tracer equations, and momentum equations, the momentum balances and surface forcing on the dry cells and cells across wet and dry area will be discussed. The implementation of W/D in NEMO v3.4.1 will also be discussed.

**Flux limiter:**

In NEMO, the water column can be considered as a control volume V. The water exchange in the intertidal zone mostly happens in the lateral directions. For such a control volume V, we can define a flux limiter as following:

The overshooting flux is

|  |  |
| --- | --- |
|  | (1) |

To preserve the positivity of water mass at each cell, the limiter coefficient *CL* must satisfy

|  |  |
| --- | --- |
|  | (2) |

So,

|  |  |
| --- | --- |
|  | (3) |

If

No flux limiter will be needed, and Eq. 3 simply becomes

|  |  |
| --- | --- |
|  |  |

Where *δt* is the time step, *q* is the source rate from precipitation and river inflow, *Fi*  is water flux rate across each boundary of the control volume, which is positive for outflow and negative for inflow. The subscript *i* identifies each boundary in the west, east, north, and south directions. *Dmin* is the minimal water depth to be kept in a dry cell while *D* is the total depth of the cell. *V* is the volume of the water column. *Sign*() is the Fortran sign function. Obviously, *CL* has the following property:

In other words, the strength of limiter varies from 0 to 1 which corresponding to fully shutting off the outflow and no limiter on outflow respectively.

The limited flux (only in the horizontal directions) has the following form

|  |  |
| --- | --- |
|  | (4) |
|  |  |

Eq. (4) is applied to all the continuity equation and tracer equations. The resultant water column on each grid cell will have its depth greater than or equal to the predefined minimal depth *Dmin*.

**Continuity Equation and Tracer Equations.**

The flux limiter is first calculated and applied to the continuity equation. We need to be careful for the explicit time splitting option for SPG because that there are many external time steps within each internal time step. In NEMO vn3.5 and earlier versions, the continuity equation is updated with the internal time step but only for the corrections for 3-D momentum trend. The split external time steps updates the bottom friction and Coriolis forcing and sea surface height gradient, which can be neglected in the dried and drying or wetting (DDW) cells. So, the current method only applies Eq. (4) to the 3-D continuity equations.

A few iterations will be needed to take into account of the interactions between two neighbouring cells when the limiter coefficient (Eq. 3) is calculated. As this flux limiter only effects around the W/D front, the iteration should converge very quickly. Also, the iteration is only needed for the continuity equation and should not be a much extra burden of the whole computation.

For the tracer Equation, Eq. 4 can simply be implemented by applying the flux limiter coefficient to the horizontal velocities. In such way, we don’t need change anything in the original source code except update the velocities.

**Momentum equations:**

For the momentum equations, the cell water depth is just the weight averaged depth of the two neighbouring water depths. And momentum itself doesn’t require the positivity. So, we won’t apply any momentum flux limiter in this method. The critical issue is how to maintain the model stability. Considering the DDW cells only have very shallow water depth, the main balance will be between the gravity forcing and bottom frictions.

For the gravity forcing, the baroclinic pressure gradient is neglected at the DDW cells while the barotropic pressure gradient is calculated in the following way:

Any momentum grid cell which is between two dried grid cells, velocity is always zero. This is similar as the land grid cells.

Any momentum grid cell which is between a wet grid cell and a dry grid cell, will have modified barotropic pressure gradients as demonstrated in the following two equations.

|  |  |
| --- | --- |
|  | (5) |

|  |  |
| --- | --- |
|  | (6) |

|  |  |
| --- | --- |
|  | (7) |

|  |  |
| --- | --- |
|  | (8) |

Where, *sign()* is the Fortran sign function. *D* is total depth, *ζ* is the sea surface height. *g* is gravity acceleration. *P* is the barotropic pressure gradient.

The bottom friction on DDW will automatically increase when the “log layer” is used as the depth there is usually very shallow. The vertical viscous/diffusive coefficient on DDW will be increased manually to keep the model stability.

The surface wind stress and surface heating/cooling will be neglected at DDW cells. The justification is that these cells are thought fully or nearly dried.

**Code implementation**:

One problem with to implement the current method is the sigma coordinate redistribution with time. The existing NEMO code uses coefficient “mut(uvf)(:,:,:)” which is based on the physical initial vertical length scale factors. In wetting/drying case, people might encounter zero depth in some grid cells at the initiation stage. So, it is modified to use the initial sigma coordinate vertical scale to do the redistribution.

The current method tries to be independent of the existing code as much as possible, that is to say, to minimise the interferences from the wetting/drying module to the existing code. CPP method seems to be the most ideal one. But we were reminded recently that NEMO development policy is trying to reduce the use of CPP. So, the code is implemented with some extra coefficients which have value of 1 or zero when applied to the non-wetting/drying case. In some other parts, the wetting/drying code was added in some IF or CASE blocks.