



Technical and numerical doc

Release 1.1

**S.Jullien, M.Caillaud, R. Benshila, L. Bordois ,
G. Cambon, F. Dumas, S. Le Gentil, F. Lemarié ,
P. Marchesiello, and S. Theetten**

Dec 05, 2022

CONTENTS

1	Primitive Equations	3
1.1	Equations in Cartesian coordinates	3
1.2	Equations in terrain following coordinates	4
2	Quasi-Hydrostatique Equations	7
2.1	Equations in Cartesian coordinate	7
3	Wave-averaged Equations	9
3.1	Equations in Cartesian coordinates	9
3.2	Embedded wave model	10
3.3	Breaking acceleration and bottom streaming	11
3.4	Formulation of wave energy dissipation	12
4	Non-Hydrostatic, Non-Boussinesq Equations	15
5	Model variables	17
5.1	Domain variables (<i>grid.h</i>)	17
5.2	Barotropic variables (<i>ocean2d.h</i>)	18
5.3	Tri-dimensionnal variables (<i>ocean3d.h</i>)	18
5.4	Surface forcing (<i>forces.h</i>)	19
6	Grid and Coordinates	21
6.1	Vertical Grid parameters	21
6.2	Wetting-Drying	24
7	Numerics	25
7.1	Overview	25
7.2	Time Stepping	26
7.3	Advection Schemes	30
7.4	Pressure gradient	36
7.5	Equation of State	36
7.6	Wetting and Drying	36
7.7	Non-Boussinesq Solver	37
8	Parametrizations	39
8.1	Vertical mixing parametrizations	39
8.2	Bottom friction	44
9	Parallelisation	45
9.1	Parallel strategy overview	45
9.2	Loops and indexes	48
9.3	Exchanges	50
10	Atmospheric Surface Boundary Layer	51

11	Open boundaries conditions	53
11.1	OBC	53
11.2	Sponge Layer	54
11.3	Nudging layers	54
11.4	Lateral forcing	55
12	Rivers	57
13	Tides	59
14	Nesting Capabilities	61
15	Sediment and Biology models	63
15.1	Bottom Boundary Layer model	63
15.2	Sediment models	66
15.3	Biogeochemical models	71
15.4	Lagrangian floats	72
16	Coupling CROCO with other models	73
16.1	OASIS philosophy	73
16.2	Detailed OASIS implementation	77
16.3	Coupled variables	81
16.4	Grids	85
17	I/O and Online Diagnostics	87
18	Review of test cases	89
18.1	Basin	89
18.2	Canyon	89
18.3	Equator	91
18.4	Inner Shelf	93
18.5	River Runoff	95
18.6	Plume	95
18.7	Gravitational/Overflow	96
18.8	Seamount	97
18.9	Shelf front	98
18.10	Equatorial Rossby Wave	99
18.11	Thacker	100
18.12	Upwelling	100
18.13	Baroclinic Vortex	101
18.14	Internal Tide	103
18.15	Internal Tide (COMODO)	104
18.16	Baroclinic Jet	106
18.17	Plannar Beach	108
18.18	Rip Current	110
18.19	Sandbar	112
18.20	Swash	113
18.21	Tank	116
18.22	Acoustic wave	117
18.23	Gravitational Adjustment	117
18.24	Internal Soliton	118
18.25	Kevlin-Helmoltz Instability	119
18.26	Horizontal tracer advection	120
19	Appendices	123
19.1	cppdefs.h	123
19.2	croco.in	132
19.3	Comparison of ROMS and CROCO versions	138

Related CPP options:

SOLVE3D	solve 3D primitive equations
UV_COR	Activate Coriolis terms
UV_ADV	Activate advection terms
NBQ	Activate non-boussinesq option
CROCO_QH	Activate quasi-hydrostatique option
MRL_WCI	Activate wave-current interactions

Preselected options:

```
# define SOLVE3D
# define UV_COR
# define UV_ADV
# undef NBQ
# undef CROCO_QH
# undef MRL_WCI
```

Presentation

By default (#undef NBQ), CROCO solves the primitive equations as in ROMS, from which it inherited the robustness and efficiency of its time-splitting implementation (Shchepetkin & McWilliams, 2005; Debreu et al., 2012) and the NBQ option proposes an extension for nonhydrostatic applications. In CROCO’s time-splitting algorithm, the ”slow mode” is similar to ROMS internal (baroclinic) mode described in Shchepetkin & McWilliams (2005), whereas, the ”fast mode” can include, in addition to the external (barotropic) mode, the pseudo-acoustic mode that allows computation of the nonhydrostatic pressure within a non-Boussinesq approach (Auclair et al., 2018). In this case, the slow internal mode is also augmented by a prognostic equation of vertical velocity, replacing the hydrostatic equation. Another option (CROCO_QH) extends the PE equations to form the quasi-hydrostatic equations, relaxing the hypothesis of weak horizontal Coriolis force (Marshall et al., 1997), thus adding a non-hydrostatic pressure component that is solved diagnostically. Then another option (MRL_WCI) treats the wave-averaged equations (McWilliams et al., 2004) with wave-current interaction terms that are both conservative and non-conservative (needing parametrizations).

PRIMITIVE EQUATIONS

At resolutions larger than 1 km (more marginally above 100 m), The ocean is a fluid that can be described to a good approximation by the primitive equations. The PE equations are simplifications from the Navier-Stokes equations made from scale considerations, along with a nonlinear equation of state, which couples the two active tracers (temperature and salinity):

- Hydrostatic hypothesis: the vertical momentum equation is reduced to a balance between the vertical pressure gradient and the buoyancy force (nonhydrostatic processes such as convection must be parametrized)
- Boussinesq hypothesis: density variations are neglected except in their contribution to the buoyancy force
- Incompressibility hypothesis (steming from the former): the three dimensional divergence of the velocity vector is assumed to be zero.
- Spherical earth approximation: the geopotential surfaces are assumed to be spheres so that gravity (local vertical) is parallel to the earth's radius
- Thin-shell approximation: the ocean depth is neglected compared to the earth's radius
- Turbulent closure hypothesis: the turbulent fluxes (which represent the effect of small scale processes on the large-scale) are expressed in terms of large-scale features

By default (#undef NBQ), CROCO solves the primitive equations. But it has also the ability to relax the first 3 hypothesis (#define NBQ). When SOLV3D is not activated, CROCO can be used as a classical shallow water model.

1.1 Equations in Cartesian coordinates

- The momentum balance in zonal x and meridional y directions, written in terms of grid-scale (resolved) and subgrid-scale velocity components:

$$\begin{aligned}\frac{\partial u}{\partial t} + \vec{\nabla} \cdot (\vec{v}u) - fv &= -\frac{\partial \phi}{\partial x} + \mathcal{F}_u + \mathcal{D}_u \\ \frac{\partial v}{\partial t} + \vec{\nabla} \cdot (\vec{v}v) + fu &= -\frac{\partial \phi}{\partial y} + \mathcal{F}_v + \mathcal{D}_v\end{aligned}$$

Turbulent closure schemes are applied to parametrized subgrid-scale vertical fluxes.

- The time evolution of a scalar concentration field, $C(x, y, z, t)$ (e.g. salinity, temperature, or nutrients), is governed by the advective-diffusive equation :

$$\frac{\partial C}{\partial t} + \vec{\nabla} \cdot (\vec{v}C) = \mathcal{F}_C + \mathcal{D}_C$$

- The equation of state is given by :

$$\rho = \rho(T, S, P)$$

- In the Boussinesq approximation, density variations are neglected in the momentum equations except in their contribution to the buoyancy force in the vertical momentum equation. Under the hydrostatic approximation, it is further assumed that the vertical pressure gradient balances the buoyancy force :

$$\frac{\partial \phi}{\partial z} = -\frac{\rho g}{\rho_0}$$

- The final equation expresses the continuity equation. For an incompressible fluid (Boussinesq approximation):

$$\vec{\nabla} \cdot \vec{v} = \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} = 0$$

The variables used are :

$\mathcal{D}_u, \mathcal{D}_v, \mathcal{D}_C$: diffusive terms

$\mathcal{F}_u, \mathcal{F}_v, \mathcal{F}_C$: forcing terms

$f(x, y)$: Traditional Coriolis parameter $2\Omega \sin \phi$

g : acceleration of gravity

$\phi(x, y, z, t)$: dynamic pressure $\phi = P/\rho_0$, with P the total pressure

$\rho_0 + \rho(x, y, z, t)$: total in situ density

u, v, w : the (x,y,z) components of vector velocity \vec{v}

1.2 Equations in terrain following coordinates

We first introduce a generalized stretched vertical coordinate system (s), which sets the variable bottom flat at $z = -h(x, y)$. s spans the range from -1 (bottom) to 0 (surface) and the transformation rules are:

$$\begin{aligned} \left(\frac{\partial}{\partial x} \right)_z &= \left(\frac{\partial}{\partial x} \right)_s - \left(\frac{1}{H_z} \right) \left(\frac{\partial z}{\partial x} \right)_s \frac{\partial}{\partial s} \\ \left(\frac{\partial}{\partial y} \right)_z &= \left(\frac{\partial}{\partial y} \right)_s - \left(\frac{1}{H_z} \right) \left(\frac{\partial z}{\partial y} \right)_s \frac{\partial}{\partial s} \\ \frac{\partial}{\partial z} &= \left(\frac{\partial s}{\partial z} \right) \frac{\partial}{\partial s} = \frac{1}{H_z} \frac{\partial}{\partial s} \end{aligned}$$

where $H_z \equiv \frac{\partial z}{\partial s}$

The vertical velocity in s coordinate is:

$$\begin{aligned} \Omega(x, y, s, t) &= \frac{1}{H_z} \left[w - (1 + s) \frac{\partial \zeta}{\partial t} - u \frac{\partial z}{\partial x} - v \frac{\partial z}{\partial y} \right] \\ w &= \frac{\partial z}{\partial t} + u \frac{\partial z}{\partial x} + v \frac{\partial z}{\partial y} + \Omega H_z \end{aligned}$$

$\Omega = 0$ at both surface and bottom.

Next, the requirement for a laterally variable grid resolution can also be met, for suitably smooth domains, by introducing an appropriate orthogonal coordinate transformation in the horizontal. Let the new coordinates be $\xi(x, y)$ and $\eta(x, y)$ where the relationship of horizontal arc length to the differential distance is given by:

$$\begin{aligned} (ds)_\xi &= \left(\frac{1}{m} \right) d\xi \\ (ds)_\eta &= \left(\frac{1}{n} \right) d\eta \end{aligned}$$

Here $m(\xi, \eta)$ and $n(\xi, \eta)$ are the scale factors which relate the differential distances $(\Delta\xi, \Delta\eta)$ to the actual (physical) arc lengths.

$$\begin{aligned}
 & \frac{\partial}{\partial t} \left(\frac{H_z u}{mn} \right) + \frac{\partial}{\partial \xi} \left(\frac{H_z u^2}{n} \right) + \frac{\partial}{\partial \eta} \left(\frac{H_z uv}{m} \right) + \frac{\partial}{\partial s} \left(\frac{H_z u \Omega}{mn} \right) \\
 & \quad - \left\{ \left(\frac{f}{mn} \right) + v \frac{\partial}{\partial \xi} \left(\frac{1}{n} \right) - u \frac{\partial}{\partial \eta} \left(\frac{1}{m} \right) \right\} H_z v = \\
 & \quad - \left(\frac{H_z}{n} \right) \left(\frac{\partial \phi}{\partial \xi} + \frac{g\rho}{\rho_o} \frac{\partial z}{\partial \xi} + g \frac{\partial \zeta}{\partial \xi} \right) + \frac{H_z}{mn} (\mathcal{F}_u + \mathcal{D}_u) \\
 & \frac{\partial}{\partial t} \left(\frac{H_z v}{mn} \right) + \frac{\partial}{\partial \xi} \left(\frac{H_z uv}{n} \right) + \frac{\partial}{\partial \eta} \left(\frac{H_z v^2}{m} \right) + \frac{\partial}{\partial s} \left(\frac{H_z v \Omega}{mn} \right) \\
 & \quad + \left\{ \left(\frac{f}{mn} \right) + v \frac{\partial}{\partial \xi} \left(\frac{1}{n} \right) - u \frac{\partial}{\partial \eta} \left(\frac{1}{m} \right) \right\} H_z u = \\
 & \quad - \left(\frac{H_z}{m} \right) \left(\frac{\partial \phi}{\partial \eta} + \frac{g\rho}{\rho_o} \frac{\partial z}{\partial \eta} + g \frac{\partial \zeta}{\partial \eta} \right) + \frac{H_z}{mn} (\mathcal{F}_v + \mathcal{D}_v) \\
 & \frac{\partial}{\partial t} \left(\frac{H_z T}{mn} \right) + \frac{\partial}{\partial \xi} \left(\frac{H_z u T}{n} \right) + \frac{\partial}{\partial \eta} \left(\frac{H_z v T}{m} \right) + \frac{\partial}{\partial s} \left(\frac{H_z \Omega T}{mn} \right) = \frac{H_z}{mn} (\mathcal{F}_T + \mathcal{D}_T) \\
 & \frac{\partial}{\partial t} \left(\frac{H_z S}{mn} \right) + \frac{\partial}{\partial \xi} \left(\frac{H_z u S}{n} \right) + \frac{\partial}{\partial \eta} \left(\frac{H_z v S}{m} \right) + \frac{\partial}{\partial s} \left(\frac{H_z \Omega S}{mn} \right) = \frac{H_z}{mn} (\mathcal{F}_S + \mathcal{D}_S) \\
 & \quad \rho = \rho(T, S, P) \\
 & \quad \frac{\partial \phi}{\partial s} = - \left(\frac{g H_z \rho}{\rho_o} \right) \\
 & \frac{\partial}{\partial t} \left(\frac{H_z}{mn} \right) + \frac{\partial}{\partial \xi} \left(\frac{H_z u}{n} \right) + \frac{\partial}{\partial \eta} \left(\frac{H_z v}{m} \right) + \frac{\partial}{\partial s} \left(\frac{H_z \Omega}{mn} \right) = 0
 \end{aligned}$$

QUASI-HYDROSTATIQUE EQUATIONS

In oceanography, traditional approximation (TA) takes the Coriolis force only partially into account by neglecting the components proportional to the cosine of latitude: $\tilde{f} = 2\Omega \cos\phi$ (see Gerkema et al., 2008, for a review). The justification for the TA is in the hypothesis that the depth of the oceans is very thin compared to the radius of the Earth. The vertical motions must then be much weaker than the horizontal ones, rendering the non-traditional (NT) Coriolis terms (with \tilde{f}) insignificant compared to the traditional terms (with f) and rendering the pressure field nearly hydrostatic. Similarly, strong vertical stratification in density, which suppresses vertical motions, also diminishes the role of NT terms. However, this argument becomes weak near the equator ($\tilde{f} \gg f$), or in motions with a strong vertical component (e.g., convection).

Note also that the QH momentum equations are shown to be more dynamically consistent than PE hydrostatic equations and that they correctly imply conservation laws for energy, angular momentum, and potential vorticity

2.1 Equations in Cartesian coordinate

- The momentum balance in x and y directions is extended to include \tilde{f} terms (zonal u component):

$$\begin{aligned} \frac{\partial u}{\partial t} + \vec{\nabla} \cdot (\vec{v}u) - fv + \tilde{f}w &= -\frac{\partial \phi}{\partial x} + \mathcal{F}_u + \mathcal{D}_u \\ \frac{\partial v}{\partial t} + \vec{\nabla} \cdot (\vec{v}v) + fu &= -\frac{\partial \phi}{\partial y} + \mathcal{F}_v + \mathcal{D}_v \end{aligned}$$

- Under the QH approximation, the quasi-hydrostatic balance is used for the vertical momentum equation, where the zonal flow partially balances the pressure gradient :

$$\frac{\partial \phi}{\partial z} = -\frac{\rho g}{\rho_0} + \tilde{f}u$$

In practice, the non-traditional term $\tilde{f}u$ is introduced as a correction to density (in the density computation subroutine rho_eos).

The variables used are :

$\mathcal{D}_u, \mathcal{D}_v$: diffusive terms

$\mathcal{F}_u, \mathcal{F}_v$: forcing terms

$f(x, y)$: Traditional Coriolis parameter $2\Omega \sin\phi$

$\tilde{f}(x, y)$: Non-traditional Coriolis parameter $2\Omega \cos\phi$

g : acceleration of gravity

$\phi(x, y, z, t)$: dynamic pressure $\phi = P/\rho_0$, with P the total pressure

$\rho_0 + \rho(x, y, z, t)$: total in situ density

u, v, w : the (x,y,z) components of vector velocity \vec{v}

WAVE-AVERAGED EQUATIONS

MRL_WCI	Activate wave-current interactions
MRL_CEW	Activate current effect on waves (2-way interaction)
WAVE_FRICTION	Activate bottom friction for WKB model and WAVE_STREAMING
WAVE_STREAMING	Activate bottom streaming (needs WAVE_FRICTION)
WAVE_BODY_STREAMING	Activate bottom streaming as body force
WAVE_OFFLINE	Activate wave forcing from offline model/data
WKB_WWAVE	Activate CROCO's monochromatic (WKB) model
OW_COUPLING	Activate coupling with spectral wave model (WW3)

Preselected options:

```
# undef WAVE_BODY_STREAMING
```

A vortex-force formalism for the interaction of surface gravity waves and currents is implemented in CROCO (Marchesiello et al., 2015; Uchiyama et al., 2010). Eulerian wave-averaged current equations for mass, momentum, and tracers are included based on an asymptotic theory by McWilliams et al. (2004) plus non-conservative wave effects due to wave breaking, associated surface roller waves, bottom streaming, and wave-enhanced vertical mixing and bottom drag especially for coastal and nearshore applications. The wave information is provided by either a spectrum-peak WKB wave-refraction model that includes the effect of currents on waves, or, alternatively, a spectrum-resolving wave model (e.g., WAVEWATCH3) can be used. In nearshore applications, the currents' cross-shore and vertical structure is shaped by the wave effects of near-surface breaker acceleration, vertical component of vortex force, and wave-enhanced pressure force and bottom drag.

3.1 Equations in Cartesian coordinates

In the Eulerian wave-averaged current equations, terms for the wave effect on currents (WEC) are added to the primitive equations. Three new variables are defined:

$$\begin{aligned}\xi^c &= \xi + \hat{\xi} \\ \phi^c &= \phi + \hat{\phi} \\ \vec{v}_L &= \vec{v} + \vec{v}_S\end{aligned}$$

where ξ^c is a composite sea level, ϕ^c absorbs the Bernoulli head $\hat{\phi}$, \vec{v}_L is the wave-averaged Lagrangian velocity, sum of Eulerian velocity and Stokes drift \vec{v}_S . The 3D Stokes velocity is non-divergent and defined for a monochromatic wave field (amplitude A , wavenumber vector $\vec{k} = (k_x, k_y)$, and frequency σ) by:

$$\begin{aligned}u_S &= \frac{A^2\sigma}{2\sinh^2(kD)} \cosh(2k(z+h)) k_x \\ v_S &= \frac{A^2\sigma}{2\sinh^2(kD)} \cosh(2k(z+h)) k_y \\ w_S &= -\int_{-h}^z \left(\frac{\partial u_S}{\partial x} + \frac{\partial v_S}{\partial y} \right) dz'\end{aligned}$$

Where $D = h + \xi^c$. The quasi-static sea level and Bernouilli head are:

$$\hat{\xi} = -\frac{A^2 k}{2 \sinh(2kD)}$$

$$\hat{\phi} = \frac{A^2 \sigma}{4k \sinh^2(kD)} \int_{-h}^z \frac{\partial^2 \vec{\mathbf{k}} \cdot \vec{\mathbf{v}}}{\partial z'^2} \sinh(2k(z - z')) dz'$$

The primitive equations become (after re-organizing advection and vortex force terms):

$$\frac{\partial u}{\partial t} + \vec{\nabla} \cdot (\vec{\mathbf{v}}_L u) - f v_L = -\frac{\partial \phi^c}{\partial x} + \left(u_S \frac{\partial u}{\partial x} + v_S \frac{\partial v}{\partial x} \right) + \mathcal{F}_u + \mathcal{D}_u + \mathcal{F}^{\mathcal{W}}_u$$

$$\frac{\partial v}{\partial t} + \vec{\nabla} \cdot (\vec{\mathbf{v}}_L v) + f u_L = -\frac{\partial \phi^c}{\partial y} + \left(u_S \frac{\partial u}{\partial y} + v_S \frac{\partial v}{\partial y} \right) + \mathcal{F}_v + \mathcal{D}_v + \mathcal{F}^{\mathcal{W}}_v$$

$$\frac{\partial \phi^c}{\partial z} + \frac{\rho g}{\rho_0} = \vec{\mathbf{v}}_S \cdot \frac{\partial \vec{\mathbf{v}}}{\partial z}$$

$$\frac{\partial C}{\partial t} + \vec{\nabla} \cdot (\vec{\mathbf{v}}_L C) = \mathcal{F}_C + \mathcal{D}_C + \mathcal{F}^{\mathcal{W}}_C$$

$$\vec{\nabla} \cdot \vec{\mathbf{v}}_L = 0$$

$$\rho = \rho(T, S, P)$$

The variables used are :

$\mathcal{D}_u, \mathcal{D}_v, \mathcal{D}_C$: diffusive terms (including wave-enhanced bottom drag and mixing)

$\mathcal{F}_u, \mathcal{F}_v, \mathcal{F}_C$: forcing terms

$\mathcal{F}^{\mathcal{W}}_u, \mathcal{F}^{\mathcal{W}}_v, \mathcal{F}^{\mathcal{W}}_C$: wave forcing terms (bottom streaming, breaking acceleration)

$f(x, y)$: Traditional Coriolis parameter $2\Omega \sin \phi$

g : acceleration of gravity

$\phi(x, y, z, t)$: dynamic pressure $\phi = P/\rho_0$, with P the total pressure

$\rho_0 + \rho(x, y, z, t)$: total in situ density

u, v, w : the (x,y,z) components of vector velocity $\vec{\mathbf{v}}$

3.2 Embedded wave model

WKB_WWAVE	Activate WKB wave model
WAVE_ROLLER	Activate wave rollers
WAVE_FRICTION	Activate bottom friction
WKB_ADD_DIFF	Activate additional diffusion to wavenumber field
MRL_CEW	Active current effect on waves
WKB_KZ_FILTER	Activate space filter on ubar,vbar,zeta for CEW
WKB_TIME_FILTER	Activate time filter on ubar,vbar,zeta for CEW
WAVE_RAMP	Activate wave ramp
ANA_BRY_WKB	Read boundary data from croco.in
WKB_OBC_WEST	Offshore wave forcing at western boundary
WKB_OBC_EAST	Offshore wave forcing at eastern boundary

Preselected options:

```
# ifdef MRL_CEW
# undef WKB_KZ_FILTER
# undef WKB_TIME_FILTER
# endif
```

(continues on next page)

(continued from previous page)

```
# define WKB_ADD_DIFF
# if defined SHOREFACE || defined SANDBAR || (defined RIP && !defined BISCA)
#   define ANA_BRY_WKB
# endif
```

A WKB wave model for monochromatic waves is embedded in CROCO following Uchiyama et al. (2010). It is based on the conservation of wave action $\mathcal{A} = E/\sigma$ and wavenumber \mathbf{k} – wave crest conservation – and is particularly suitable for nearshore beach applications, allowing refraction from bathymetry and currents (but no diffraction or reflection), with parametrizations for wave breaking and bottom drag:

$$\begin{aligned} \frac{\partial \mathcal{A}}{\partial t} + \vec{\nabla} \cdot \mathcal{A} \vec{c}_g &= -\frac{\epsilon^w}{\sigma} \\ \frac{\partial \vec{\mathbf{k}}}{\partial t} + \vec{c}_g \cdot \nabla \vec{\mathbf{k}} &= -\vec{\mathbf{k}} \cdot \nabla \vec{\mathbf{V}} - \frac{k\sigma}{\sinh 2kD} \nabla D \end{aligned}$$

$\vec{\mathbf{V}}$ is the depth-averaged velocity vector and σ is the intrinsic frequency defined by the linear dispersion relation $\sigma^2 = gk \tanh kD$. Current effects on waves are noticeable in the groupe velocity c_g which gets two components: the doppler shift due to currents on waves and the groupe velocity of the primary carrier waves :

$$\vec{c}_g = \vec{\mathbf{V}} + \frac{\sigma}{2k^2} \left(1 + \frac{2kD}{\sinh 2kD} \right) \vec{\mathbf{k}}$$

The currents may need filtering before entering the wave model equations because the current field should evolve slowly with respect to waves in the asymptotic regime described by McWilliams et al. (2004). By default, this filtering is turned off (WKB_KZ_FILTER, WKB_TIME_FILTER).

ϵ^w is the depth-integrated rate of wave energy dissipation due to depth-induced breaking ϵ^b (including white capping) and bottom friction ϵ^{wd} , both of which must be parameterized (in WKB, WW3 or CROCO if defined WAVE_OFFLINE):

$$\epsilon^w = \epsilon^b + \epsilon^{wd}$$

3.3 Breaking acceleration and bottom streaming

A formulation for ϵ^b is needed in both the wave model (dissipation term) and the circulation model (acceleration term). In the wave-averaged momentum equations of the circulation model, the breaking acceleration enters as a body force through \mathcal{F}^W :

$$\vec{\mathbf{F}}^b = \frac{\epsilon^b}{\rho\sigma} \vec{\mathbf{k}} f_b(z)$$

where $f_b(z)$ is a normalized vertical distribution function representing vertical penetration of momentum associated with breaking waves from the surface. The penetration depth is controlled by a vertical length-scale taken as H_{rms} .

The wave model can also include a roller model with dissipation ϵ^r . In this case:

$$\vec{\mathbf{F}}^b = \frac{(1 - \alpha_r)\epsilon^b + \epsilon^r}{\rho\sigma} \vec{\mathbf{k}} f_b(z)$$

The idea is that some fraction α_r of wave energy is converted into rollers that propagate toward the shoreline before dissipating, while the remaining fraction $1 - \alpha_r$ causes local dissipation (hence current acceleration). It can be useful for correcting ϵ^b with some flexibility to depict different breaking wave and beach forms (e.g., spilling or plunging breakers, barred or plane beaches), although the parameter B_b can also be used for that. See Uchiyama et al. (2010) for the roller equation and ϵ^r formulation.

Wave-enhanced bottom dissipation enters in the momentum equations through a combined wave-current drag formulation (see parametrizations) and bottom streaming. The latter is due to dissipation of wave energy in the

wave boundary layer that causes the instantaneous, oscillatory wave bottom orbital velocities to be slightly in phase from quadrature; this causes a wave stress (bottom streaming) in the wave bottom boundary layer along the direction of wave propagation (Longuet-Higgins, 1953). The effect of bottom streaming in momentum balance is accounted for by using the wave dissipation due to bottom friction with an upward decaying vertical distribution:

$$\vec{\mathbf{F}}^{st} = \frac{\epsilon^{wd}}{\rho\sigma} \vec{\mathbf{k}} f_{st}(z)$$

where $f_{st}(z)$ is a vertical distribution function.

3.4 Formulation of wave energy dissipation

WAVE_SFC_BREAK	Activate surface breaking acceleration
WAVE_BREAK_CT93	Activate Church & Thornton (1993) breaking acceleration (default)
WAVE_BREAK_TG86	Activate Thornton & Guza (1983, 1986)

Preselected options:

```
# define WAVE_BREAK_CT93
# undef WAVE_BREAK_TG86
# undef WAVE_SFC_BREAK
```

While a few formulations for ϵ^b are implemented in CROCO, the one by Church and Thornton (1993) is generally successful for nearshore beach applications:

$$\epsilon^b = \frac{3}{16} \sqrt{\pi} \rho g B_b^3 \frac{H_{rms}^3}{D} \left\{ 1 + \tanh \left[8 \left(\frac{H_{rms}}{\gamma_b D} - 1 \right) \right] \right\} \left\{ 1 - \left[1 + \left(\frac{H_{rms}}{\gamma_b D} \right)^2 \right]^{-2.5} \right\}$$

where B_b and γ_b are empirical parameters related to wave breaking. γ_b represents the wave height-to-depth ratio for which all waves are assumed to be breaking and B_b is the fraction of foam on the face, accounting for the type of breaker. H_{rms} is the RMS wave height. For the DUCK94 experiment, Uchiyama et al. (2010) suggest $\gamma_b = 0.4$ and $B_b = 0.8$, while for Biscarrosse Beach, Marchesiello et al. (2015) use $\gamma_b = 0.3$ and $B_b = 1.3$ from calibration with video cameras.

For ϵ^{wd} , the dissipation caused by bottom viscous drag on the primary waves, we use a parameterization for the realistic regime of a turbulent wave boundary layer, consistent with the WKB spectrum-peak wave modeling:

$$\epsilon^{wd} = \frac{1}{2\sqrt{\pi}} \rho f_w u_{orb}^3$$

where u_{orb} is the wave orbital velocity magnitude and f_w is a wave friction factor, function of roughness length z_0 :

$$u_{orb} = \frac{\sigma H_{rms}}{2 \sinh kD}$$

$$f_w = 1.39 \left(\frac{\sigma z_0}{u_{orb}} \right)^{0.52}$$

References

Marchesiello, P.; Benshila, R.; Almar, R.; Uchiyama, Y.; McWilliams, J.C., and Shchepetkin, A., 2015. On tridimensional rip current modeling. *Ocean Model.*, 96, 36-48.

McWilliams, J.C., Restrepo, J.M., and Lane, M.R., 2004. An asymptotic theory for the interaction of waves and currents in coastal waters. *J. Fluid Mech.*, 511, 135-178.

Thornton, E.B. & R.T. Guza, 1983: Transformation of wave height distribution, *J. Geophys. Res.* 88, 5925-5938.

Uchiyama, Y., McWilliams, J., Shchepetkin, A., 2010. Wave-current interaction in an oceanic circulation model with a vortex-force formalism: application to the surf zone. *Ocean Modell.* 34, 16–35.

Weir, B., Uchiyama, Y., Lane, E. M., Restrepo, J. M., & McWilliams, J. C. (2011). A vortex force analysis of the interaction of rip currents and surface gravity waves. *Journal of Geophysical Research: Oceans*, 116(C5).

NON-HYDROSTATIC, NON-BOUSSINESQ EQUATIONS

The full set of Navier-Stokes equations for a free-surface ocean is explicitly integrated in the non-hydrostatic, non-Boussinesq version of CROCO (#define NBQ). In this approach, acoustic waves are solved explicitly to avoid Boussinesq-degeneracy, which inevitably leads to a 3D Poisson-system in non-hydrostatic Boussinesq methods – detrimental to computational costs and difficult to implement within a split-explicit barotropic/baroclinic model.

NBQ equations include the momentum and continuity equations, the surface kinematic relation (for free surface), temperature, salinity – or other tracer C – and the equation of state, which reads in Cartesian coordinates:

$$\begin{aligned}
 \frac{\partial \rho u}{\partial t} + \vec{\nabla} \cdot (\rho \vec{v} u) - \rho f v - \rho \tilde{f} w &= -\frac{\partial P}{\partial x} + \lambda \frac{\partial \vec{\nabla} \cdot \vec{v}}{\partial x} + \mathcal{F}_u + \mathcal{D}_u \\
 \frac{\partial \rho v}{\partial t} + \vec{\nabla} \cdot (\rho \vec{v} v) + \rho f u &= -\frac{\partial P}{\partial y} + \lambda \frac{\partial \vec{\nabla} \cdot \vec{v}}{\partial y} + \mathcal{F}_v + \mathcal{D}_v \\
 \frac{\partial \rho w}{\partial t} + \vec{\nabla} \cdot (\rho \vec{v} w) - \rho \tilde{f} u &= -\frac{\partial P}{\partial z} - \rho g + \lambda \frac{\partial (\vec{\nabla} \cdot \vec{v})}{\partial z} + \mathcal{F}_w + \mathcal{D}_w \\
 \frac{\partial \rho}{\partial t} &= -\vec{\nabla} \cdot (\rho \vec{v}) \\
 \frac{\partial \xi}{\partial t} &= w_f|_{z=\xi} - \vec{v}|_{z=\xi} \cdot \vec{\nabla} \xi \\
 \frac{\partial \rho C}{\partial t} &= -\vec{\nabla} \cdot (\rho \vec{v} C) + \mathcal{F}_C + \mathcal{D}_C
 \end{aligned}$$

λ is the second (bulk) viscosity, associated with compressibility (it can be used to damp acoustic waves).

A relation between ρ and P is now required. To that end, and as part of a time-splitting approach, density is decomposed into slow and fast components based on a first-order decomposition with respect to total pressure. In the following, s and f subscripts refer to slow and fast-mode components respectively:

$$\begin{aligned}
 \rho &= \rho_s(T, S, P) + \overbrace{\left. \frac{\partial \rho}{\partial P} \right|_{T,S} \delta P + O(\delta P^2)}^{\rho_f = c_s^{-2} P_f} \\
 P &= \underbrace{P_{atm} + \int_z^\xi (\rho_s - \rho_0) g \, dz'}_{SLOW} + \underbrace{\rho_0 g (\xi - z) + \delta P}_{FAST}
 \end{aligned}$$

c_s is the speed of sound and $\delta P = P_f$ is the nonhydrostatic pressure.

The Navier-Stokes equations are then integrated with two different time-steps within the time-splitting approach. The slow mode is identical to ROMS whereas the fast mode (in the NBQ equations) is 3D and the fast time step

includes the integration of the compressible terms of the momentum and continuity equations. In vector form:

$$\begin{aligned}
 \frac{\partial \rho \vec{v}}{\partial t} &= \underbrace{-\vec{\nabla} \cdot (\rho \vec{v} \otimes \vec{v}) - 2\rho \vec{\Omega} \times \vec{v} - \vec{\nabla} \left(\int_z^{\xi_f} (\rho_s - \rho_0) g \, dz' \right) + \vec{\mathcal{F}}_{\vec{v}} + \vec{\mathcal{D}}_{\vec{v}}}_{\text{SLOW}} \\
 &\quad \underbrace{-\rho_0 g \vec{\nabla} \xi_f - \vec{\nabla} P_f + \rho \vec{g} + \lambda \vec{\nabla} (\vec{\nabla} \cdot \vec{v})}_{\text{FAST}} \\
 \frac{\partial \rho_f}{\partial t} &= -\frac{\partial \rho_s}{\partial t} - \vec{\nabla} \cdot (\rho \vec{v}) \\
 P_f &= c_s^2 \rho_f \\
 \frac{\partial \xi_f}{\partial t} &= w_f|_{z=\xi} - \vec{v}_f|_{z=\xi} \cdot \vec{\nabla} \xi_f \\
 \frac{\partial \rho C_s}{\partial t} &= -\vec{\nabla} \cdot (\rho \vec{v} C_s) + \mathcal{F}_C + \mathcal{D}_C \\
 \rho_s &= \rho(T_s, S_s, \xi_f) \\
 \rho &= \rho_s + \rho_f
 \end{aligned}$$

The momentum is integrated both in slow and fast modes but the right-hand-side of the equation is split in two parts: a slow part, made of slowly varying terms (advection, Coriolis force, baroclinic pressure force and viscous dissipation), and a fast part, made of fast-varying terms (the surface-induced and compressible pressure force, the weight and the dissipation associated with bulk-viscosity). This momentum equation is numerically integrated twice, once with a large time-step keeping the fast part constant, and once with a smaller time-step keeping the slow part constant. This is much more computationally efficient than integrating the whole set of equations at the same fast time step. More details can be found in Auclair et al. (2018).

Note that the solved acoustic waves can become pseudo-acoustic if their phase speed c_s is artificially slowed down (it is a model input). In this case, high-frequency processes associated with bulk compressibility may be unphysical, but a coherent solution for slow non-hydrostatic dynamics is preserved, while the CFL constraint is relaxed.

MODEL VARIABLES

Model variables are defined in .h Fortran 77 files :

5.1 Domain variables (*grid.h*)

grid.h : Environmental two-dimensional arrays associated with curvilinear horizontal coordinate system

h : Model topography (bottom depth [m] at RHO-points.)

dh : Topography increment in case of moving bathymetry

f : Coriolis parameter [1/s].

fomn : Compound term, $f/[p_m * p_n]$ at RHO points.

angler : Angle [radians] between XI-axis and the direction to the EAST at RHO-points.

latr : Latitude (degrees_north) at RHO-, U-, and V-points.

latu

latv

lonr : Longitude (degrees_east) at RHO-, U-, and V-points.

lonu

lonv

xp : XI-coordinates [m] at PSI-points.

xr : XI-coordinates [m] at RHO-points.

yp : ETA-coordinates [m] at PSI-points.

yr : ETA-coordinates [m] at RHO-points.

pm : Coordinate transformation metric “m” [1/meters] associated with the differential distances in XI.

pn : Coordinate transformation metric “n” [1/meters] associated with the differential distances in ETA.

om_u : Grid spacing [meters] in the XI -direction at U-points.

om_v : Grid spacing [meters] in the XI -direction at V-points.

on_u : Grid spacing [meters] in the ETA-direction at U-points.

on_v : Grid spacing [meters] in the ETA-direction at V-points.

dmde : ETA-derivative of inverse metric factor “m”, $d(1/M)/d(ETA)$.

dndx : XI-derivative of inverse metric factor “n”, $d(1/N)/d(XI)$.

pmon_p : Compound term, pm/pn at PSI-points.
pmon_r : Compound term, pm/pn at RHO-points.
pmon_u : Compound term, pm/pn at U-points.
pnom_p : Compound term, pn/pm at PSI-points.
pnom_r : Compound term, pn/pm at RHO-points.
pnom_v : Compound term, pn/pm at V-points.

rmask : Land-sea masking arrays at RHO-,U-,V- and PSI-points (rmask,umask,vmask) = (0=Land, 1=Sea)
umask
vmask
pmask : pmask=(0=Land, 1=Sea, 1-gamma2 =boundary).

reducu : reduction coefficient along x-axis for rivers sections
reducv : reduction coefficient along y-axis for rivers sections

5.2 Barotropic variables (*ocean2d.h*)

ocean2d.h : 2D dynamical variables for fast mode

zeta,rzeta : Free surface elevation [m] and its time tendency;
ubar,rubar : Vertically integrated 2D velocity components in
vbar,rvbar : XI- and ETA-directions and their time tendencies;

5.3 Tri-dimensionnal variables (*ocean3d.h*)

ocean2d.h : 3D tracers dynamical variables for baroclinic mode

u,v : 3D velocity components in XI- and ETA-directions
t : tracer array (temperature, salinity, passive tracers, sediment)
Hz : level thickness
z_r : depth at rho point
z_w : depth at w point
Huon : transport a U point
Hvon : transport at V point
We, Wi : vertical velocity (explicit, implicit)
rho : density anomaly
rho1 : potential density at 1 atm

5.4 Surface forcing (forces.h)

forces.h :

Surface momentum flux (wind stress) :

sustr : XI- and ETA-components of kinematic surface momentum flux

svstr : wind stresses) defined at horizontal U- and V-points.dimensioned as [m²/s²].

Bottom mometum flux :

bustr : XI- and ETA-components of kinematic bottom momentum flux

bvstr : (drag) defined at horizontal U- and V-points [m²/s²].!

Surface tracers fluxes :

stflx : Kinematic surface fluxes of tracer type variables at horizontal RHO-points. Physical dimensions [degC m/s] - temperature; [PSU m/s] - salinity.

dqdt : Kinematic surface net heat flux sensitivity to SST [m/s].

sst : Current sea surface temperature [degree Celsius].

dqdtg : Two-time-level grided data for net surface heat flux

sstg : sensitivity to SST grided data [Watts/m²/Celsius] and sea surface temperature [degree Celsius].

dqdtp : Two-time-level point data for net surface heat flux

sstp : sensitivity to SST grided data [Watts/m²/Celsius] and sea surface temperature [degree Celsius].

tsst : Time of sea surface temperature data.

sss : Current sea surface salinity [PSU].

tair : surface air temperature at 2m [degree Celsius].

wsp : wind speed at 10m [degree Celsius].

rhum : surface air relative humidity 2m [fraction]

prate : surface precipitation rate [cm day-1]

radlw : net terrestrial longwave radiation [Watts meter-2]

radsw : net solar shortwave radiation [Watts meter-2]

patm2d : atmospheric pressure above mean seal level

paref : reference pressure to compute inverse barometer effect

srlx : Kinematic surface shortwave solar radiation flux [degC m/s] at horizontal RHO-points

Wind induced waves everything is defined at rho-point :

wfrq : wind-induced wave frequency [rad/s]

uorb : xi-component of wave-induced bed orbital velocity [m/s]

vorb : eta-component of wave-induced bed orbital velocity [m/s]

wdrx : cosine of wave direction [non dimension]

wdre : sine of wave direction [non dimension]

whrm : (RMS) wave height (twice the wave amplitude) [m]

wepb : breaking dissipation rate (epsilon_b term) [m³/s³]

wepd : frictional dissipation rate (epsilon_d term) [m³/s³]

wepr : roller dissipation rate (epsilon_r term) [m³/s³]

wbst : frictional dissipation stress ($e_d k/\sigma$) [m²/s²]

Wave averaged quantities :

brk2dx : xi-direction 2D breaking dissipation (rho)
brk2de : eta-direction 2D breaking dissipation (rho)
frc2dx : xi-direction 2D frictional dissipation (rho)
frc2de : eta-direction 2D frictional dissipation (rho)
ust2d : xi-direction Stokes transport (u-point)
vst2d : eta-direction Stokes transport (v-point)
sup : quasi-static wave set-up (rho-point)
calP : pressure correction term (rho-point)
Kapsrf : Bernoulli head term at the surface (rho-point)
brk3dx : xi-direction 3D breaking dissipation (rho)
brk3de : eta-direction 3D breaking dissipation (rho)
ust : xi-direction 3D Stokes drift velocity (u-point)
vst : eta-direction 3D Stokes drift velocity (v-point)
wst : vertical 3D Stokes drift velocity (rho-point)
Kappa : 3D Bernoulli head term (rho-point)
kvf : vertical vortex force term (K term, 3D, rho-point)
Akb : breaking-wave-induced additional diffusivity (w-point)
Akw : wave-induced additional diffusivity (rho-point)
E_pre : previous time-step value for Akw estimation (rho)
frc3dx : xi-direction 3D frictional dissipation (rho)
frc3de : eta-direction 3D frictional dissipation (rho)

GRID AND COORDINATES

Related CPP options:

CURVGRID	Activate curvilinear coordinate transformation
SPHERICAL	Activate longitude/latitude grid positioning
MASKING	Activate land masking
WET_DRY	Activate wetting-Drying scheme
NEW_S_COORD	Choose new vertical S-coordinates

Preselected options:

```
# define CURVGRID
# define SPHERICAL
# define MASKING
# undef WET_DRY
# undef NEW_S_COORD
```

6.1 Vertical Grid parameters

Two vertical transformation are available for the generalized vertical terrain following vertical system : By default, we have :

$$z(x, y, \sigma, t) = z_0(x, y, \sigma) + \zeta(x, y, t) \left[1 + \frac{z_0(x, y, \sigma)}{h(x, y)} \right] \quad (6.1)$$

$$z_0(x, y, \sigma) = h_c \sigma + [h(x, y) - h_c] Cs(\sigma) \quad (6.2)$$

When activated the cpp key NEW_S_COORD, we have :

$$z(x, y, \sigma, t) = \zeta(x, y, \sigma) + [\zeta(x, y, t) + h(x, y)] z_0(x, y, \sigma) \quad (6.3)$$

$$z_0(x, y, \sigma) = \frac{h_c \sigma + h(x, y) Cs(\sigma)}{h_c + h(x, y)} \quad (6.4)$$

with :

- $z_0(x, y, \sigma)$ a nonlinear vertical transformation
- $\zeta(x, y, \sigma)$ the free-surface
- $h(x, y)$ the ocean bottom
- σ a fractional vertical stretching coordinate, $-1 \leq \sigma \leq 0$
- h_c a positive thickness controlling the stretching
- $Cs(\sigma)$ a nondimensional, monotonic, vertical stretching, $-1 \leq (C\sigma) \leq 0$

Vertical grid stretching is controlled by the following parameters, that have to be set similarly in `croco.in`, and `crocotools_param.m`:

theta_s	Vertical S-coordinate surface stretching parameter. When building the climatology and initial CROCO files, we have to define the vertical grid. Warning! The different vertical grid parameters should be identical in this <code>crocotools_param.m</code> and in <code>croco.in</code> . This is a serious cause of bug.
theta_b	Vertical S-coordinate bottom stretching parameter.
hc	Vertical S-coordinate Hc parameter. It gives approximately the transition depth between the horizontal surfacelevels and the bottom terrain following levels. (Note it should be inferior to <code>hmin</code> in case of <code>Vtransform = 1</code>).

Then we have, with N the number of vertical levels:

- with the old transformation :

$$Cs(\sigma) = (1 - \theta_b) \frac{\sinh(\theta_s \sigma)}{N} + \theta_b \left[\frac{0.5 \tanh((\sigma + 0.5) \theta_s)}{\tanh(0.5 \theta_s)} - 0.5 \right]$$

- with `NEW_S_COORD` defined :

$$sc = \frac{\sigma - N}{N} \tag{6.5}$$

$$csf = \frac{1 - \cosh(\theta_s sc)}{\cosh(\theta_s) - 1} \quad \text{if } \theta_b > 0, \quad csf = -sc^2 \quad \text{otherwise} \tag{6.6}$$

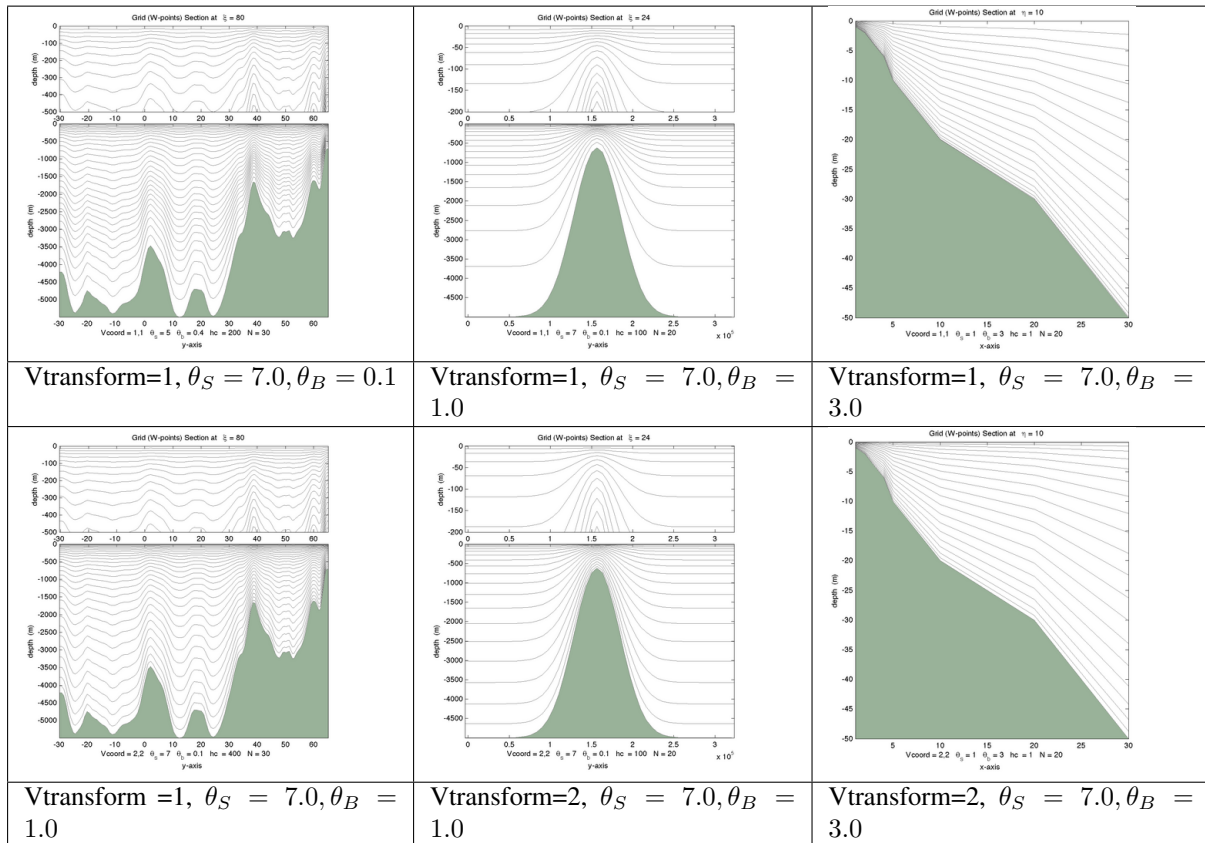
$$Cs(\sigma) = \frac{e^{\theta_b csf} - 1}{1 - e^{-\theta_b}} \quad \text{if } \theta_s > 0, \quad Cs(\sigma) = csf \quad \text{otherwise} \tag{6.7}$$

Other parameters have to be set to prepare the grid file in `crocotools_param.m`:

vtransform	S-coordinate type (1: old- ; 2: new- coordinates). It is associated to #NEW_S_COORD cpp-keys in CROCO source code.
hmin	Minimum depth in meters. The model depth is cut at this level to prevent, for example, the occurrence of model grid cells without water. This does not influence the masking routines. At lower resolution, hmin should be quite large (for example 150m for dl=1/2). Otherwise, since topography smoothing is based on, the bottom slopes can be totally eroded.
hmax_coast	Maximum depth under the mask. It prevents selected isobaths (here 500 m) to go under the mask. If this is the case, this could be a source of problems for western boundary currents (for example).
hmax	Maximum depth
rtarget	This variable control the maximum value of the σ -parameter that measures the slope of the sigma layers (Beckmann and Haidvogel, 1993): To prevent horizontal pressure gradients errors, well in terrain-following coordinate models (Haney, 1991), realistic topography requires some smoothing. Empirical results have shown that reliable model results are obtained if does not exceed 0.2.
n_filter_deep_topo	Number of pass of a Hanning filter to prevent the occurrence of noise and isolated seamounts on deep regions.
n_filter_final	Number of pass of a Hanning filter at the end of the smoothing process to be sure that no noise is present in the topography.

The effects of theta_s, theta_b, hc, and N can be tested using the Matlab script : `croco_tools/Preprocessing_tools/test_vgrid.m`

Below are some examples of different vertical choices (Courtesy of [ROMS-RUTGERS](#) team) :



6.2 Wetting-Drying

The Wetting-Drying scheme is derived from John Warner's code (Rutgers ROMS) and adapted to the time stepping scheme of CROCO. The main idea is to cancel the outgoing momentum flux (not the incoming) from a grid cell if its total depth is below a threshold value (critical depth D_{crit} between 5 and 20 cm according to local slope; D_{crit} min and max adjustable in param.h). This scheme is tested in the Thacker case producing oscillations in a rotating bowl for which an analytical solution is known.

7.1 Overview

CROCO solves the primitive equations in an Earth-centered rotating environment. It is discretized in coastline- and terrain-following curvilinear coordinates using high-order numerical methods. It is a split-explicit, free-surface ocean model, where short time steps are used to advance the surface elevation and barotropic momentum, with a much larger time step used for temperature, salinity, and baroclinic momentum.

The complete time stepping algorithm is described in Shchepetkin and McWilliams (2005); see also Soufflet et al. (2016). The model has a 2-way time-averaging procedure for the barotropic mode, which satisfies the 3D continuity equation. The specially designed 3rd order predictor-corrector time step algorithm allows a substantial increase in the permissible time-step size.

Combined with the 3rd order time-stepping, a 3rd- or 5th-order, upstream-biased horizontal advection scheme (alternatively WENO or TVD for monotonicity preservation) allows the generation of steep gradients, enhancing the effective resolution of the solution for a given grid size (Shchepetkin and McWilliams, 1998; Soufflet et al., 2016; Menesguen et al., 2018, Borges et al., 2008). Because of the implicit diffusion in upstream advection schemes, explicit lateral viscosity is not needed in CROCO for damping numerical dispersion errors.

For vertical advection, SPLINE or WENO5 schemes are proposed (besides lower-order schemes). For SPLINES (default), an option for an adaptive, Courant-number-dependent implicit scheme is proposed that has the advantage to render vertical advection unconditionally stable while maintaining good accuracy in locations with small Courant numbers (Shchepetkin, 2015). This is also available for tracers.

Tracers are treated similarly to momentum. A 3rd- or 5th-order upstream-biased horizontal advection scheme is implemented, but in regional configurations the diffusion part of this scheme is rotated along isopycnal surfaces to avoid spurious diapycnal mixing and loss of water masses (Marchesiello et al., 2009; Lemarié et al., 2012). For regional/coastal applications, a highly accurate pressure gradient scheme (Shchepetkin and McWilliams, 2003) limits the other type of errors (besides spurious diapycnal mixing) frequently associated with terrain-following coordinate models.

If a lateral boundary faces the open ocean, an active, implicit, upstream biased, radiation condition connects the model solution to the surroundings (Marchesiello et al., 2001). It comes with sponge layers for a better transition between interior and boundary solutions (explicit Laplacian diffusion and/or newtonian damping)

For nearshore problems, where waves become the dominant forcing of circulation, a vortex-force formalism for the interaction of surface gravity waves and currents is implemented in CROCO (Uchiyama et al., 2010).

CROCO can be used either as a Boussinesq/hydrostatic model, or a non-hydrostatic/non-Boussinesq model (NBQ; Auclair et al., 2018). The NBQ solver is relevant in problems from a few tens of meters to LES or DNS resolutions. It comes with shock-capturing advection schemes (WENO5, TVD) and fully 3D turbulent closure schemes (GLS, Smagorinsky).

CROCO includes a variety of additional features, e.g., 1D turbulent closure schemes (KPP, GLS) for surface and benthic boundary layers and interior mixing; wetting and drying; sediment and biological models; AGRIF interface for 2-way nesting; OASIS coupler for ocean-waves-atmosphere coupling. . .

7.2 Time Stepping

CROCO is discretized in time using a third-order predictor-corrector scheme (referred to as LFAM3) for tracers and baroclinic momentum. It is a split-explicit, free-surface ocean model, where short time steps are used to advance the surface elevation and barotropic momentum, with a much larger time step used for tracers, and baroclinic momentum. The model has a 2-way time-averaging procedure for the barotropic mode, which satisfies the 3D continuity equation. The specially designed 3rd order predictor-corrector time step algorithm is described in Shechepetkin and McWilliams (2005) and is summarized in this subsection.

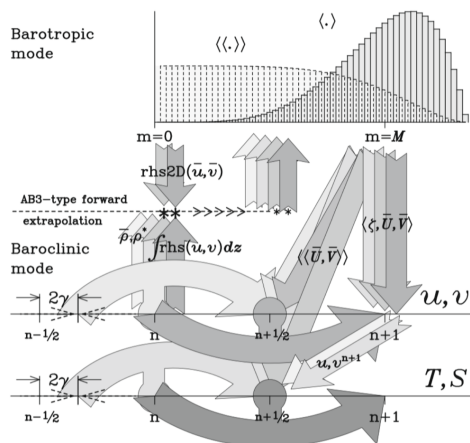


Fig. 1: Fig: schematic view of the Croco predictor-corrector hydrostatic kernel

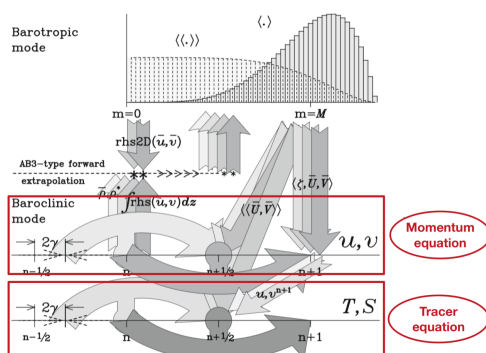
General structure of the time-stepping:

```

call prestep3D_thread()    ! Predictor step for 3D momentum and tracers
call step2d_thread()      ! Barotropic mode
call step3D_uv_thread()   ! Corrector step for momentum
call step3D_t_thread()    ! Corrector step for tracers
    
```

7.2.1 3D momentum and tracers

Predictor-corrector approach : **Leapfrog (LF) predictor with 3rd-order Adams-Moulton (AM) interpolation (LFAM3 timestepping)**. This scheme is used to integrate 3D advection, the pressure gradient term, the continuity equation and the Coriolis term which are all contained in the RHS operator.



For a given quantity q

$$\begin{cases}
 q^{n+1,*} &= q^{n-1} + 2\Delta t \text{ RHS} \{q^n\} & \text{(LF)} \\
 q^{n+\frac{1}{2}} &= \frac{5}{12} q^{n+1,*} + \frac{2}{3} q^n - \frac{1}{12} q^{n-1} & \text{(AM3)} \\
 q^{n+1} &= q^n + \Delta t \text{ RHS} \left\{q^{n+\frac{1}{2}}\right\} & \text{(corrector)}
 \end{cases}$$

which can be rewritten in a compact way as used in the Croco code :

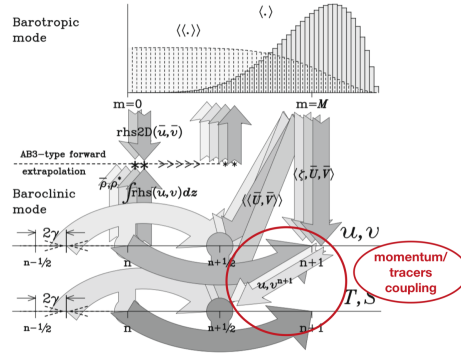
$$q^{n+\frac{1}{2}} = \left(\frac{1}{2} - \gamma\right) q^{n-1} + \left(\frac{1}{2} + \gamma\right) q^n + (1 - \gamma)\Delta t \text{ RHS } \{q^n\}$$

$$q^{n+1} = q^n + \Delta t \text{ RHS } \{q^{n+\frac{1}{2}}\}$$

with $\gamma = \frac{1}{6}$.

Physical parameterizations for vertical mixing, rotated diffusion and viscous/diffusion terms are computed once per time-step using an Euler step.

7.2.2 Tracers-momentum coupling



The numerical integration of internal waves can be studied using the following subsystem of equations

$$\begin{cases} \partial_z w + \partial_x u = 0 \\ \partial_z p + \rho g = 0 \\ \partial_t u + \frac{1}{\rho_0} \partial_x p = 0 \\ \partial_t \rho + \partial_z (w\rho) = 0 \end{cases}$$

Predictor step:

$$\partial_x p^n = g \partial_x \left(\int_z^0 \rho^n dz \right) \quad \rightarrow \quad u^{n+\frac{1}{2}} = \left(\frac{1}{2} - \gamma\right) u^{n-1} + \left(\frac{1}{2} + \gamma\right) u^n + (1 - \gamma) \frac{\Delta t}{\rho_0} (\partial_x p^n)$$

$$w^n = - \int_{-H}^z \partial_x u^n dz' \quad \rightarrow \quad \rho^{n+\frac{1}{2}} = \left(\frac{1}{2} - \gamma\right) \rho^{n-1} + \left(\frac{1}{2} + \gamma\right) \rho^n + (1 - \gamma) \Delta t \partial_z (w^n \rho^n)$$

Corrector step:

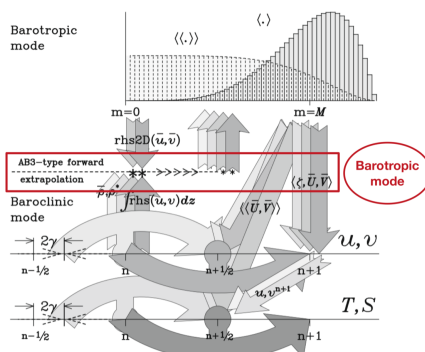
$$\partial_x p^{n+\frac{1}{2}} = g \partial_x \left(\int_z^0 \rho^{n+\frac{1}{2}} dz \right) \quad \rightarrow \quad u^{n+1} = u^n + \frac{\Delta t}{\rho_0} (\partial_x p^{n+\frac{1}{2}})$$

$$w^{n+\frac{1}{2}} = - \int_{-H}^z \partial_x \left\{ \frac{3u^{n+\frac{1}{2}}}{4} + \frac{u^n + u^{n+1}}{8} \right\} dz' \quad \rightarrow \quad \rho^{n+1} = \rho^n + \Delta t \partial_z (w^{n+\frac{1}{2}} \rho^{n+\frac{1}{2}})$$

Consequences:

- 3D-momentum integrated before the tracers in the corrector
- 2 evaluations of the pressure gradient per time-step
- 3 evaluations of the continuity equation per time-step

7.2.3 Barotropic mode



Generalized forward-backward (predictor-corrector)

1. AB3-type extrapolation

$$D^{m+\frac{1}{2}} = H + \left(\frac{3}{2} + \beta\right) \zeta^m - \left(\frac{1}{2} + 2\beta\right) \zeta^{m-1} + \beta \zeta^{m-2}$$

$$\bar{u}^{m+\frac{1}{2}} = \left(\frac{3}{2} + \beta\right) \bar{u}^m - \left(\frac{1}{2} + 2\beta\right) \bar{u}^{m-1} + \beta \bar{u}^{m-2}$$

2. Integration of ζ

$$\zeta^{m+1} = \zeta^m - \Delta\tau \partial_x (D^{m+\frac{1}{2}} \bar{u}^{m+\frac{1}{2}})$$

3. AM4 interpolation

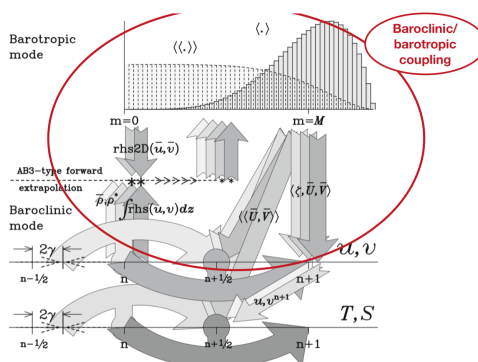
$$\zeta^* = \left(\frac{1}{2} + \gamma + 2\varepsilon\right) \zeta^{m+1} + \left(\frac{1}{2} - 2\gamma - 3\varepsilon\right) \zeta^m + \gamma \zeta^{m-1} + \varepsilon \zeta^{m-2}$$

4. Integration of \bar{u}

$$\bar{u}^{m+1} = \frac{1}{D^{m+1}} \left[D^m \bar{u}^m + \Delta\tau \text{RHS2D}(D^{m+\frac{1}{2}}, \bar{u}^{m+\frac{1}{2}}, \zeta^*) \right]$$

where the parameter values are $(\beta, \gamma, \varepsilon) = (0.281105, 0.088, 0.013)$ except when the filter_none option is activated (see below).

7.2.4 Baroclinic-barotropic coupling

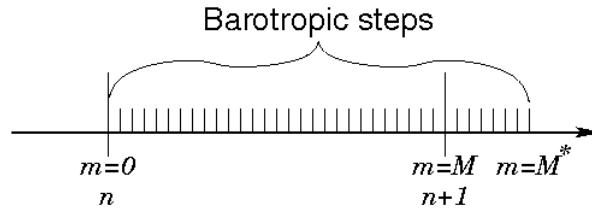


Slow forcing term of the barotrope by the barocline is extrapolated

$$\mathcal{F}_{3D}^{n+\frac{1}{2}} = \left\{ \int rhs(u, v) dz - rhs2D(\bar{u}, \bar{v}) \right\}^{n+\frac{1}{2}} = \text{Extrap}(\mathcal{F}_{3D}^n, \mathcal{F}_{3D}^{n-1}, \mathcal{F}_{3D}^{n-2})$$

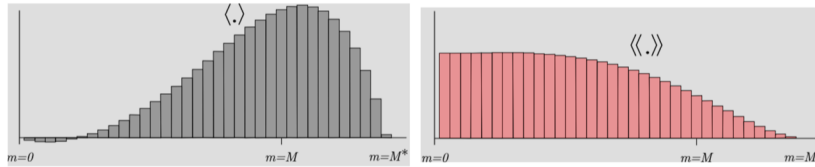
M2_FILTER_POWER option

Barotropic integration from n to $n + M^* \Delta\tau$ ($M^* \leq 1.5M$)



Because of predictor-corrector integration two barotropic filters are needed

- $\langle \zeta \rangle^{n+1} \rightarrow$ update of the vertical grid
- $\langle U \rangle^{n+1} \rightarrow$ correction of baroclinic velocities at time $n + 1$
- $\langle\langle U \rangle\rangle^{n+\frac{1}{2}} \rightarrow$ correction of baroclinic velocities at time $n + \frac{1}{2}$



M2_FILTER_NONE option

Motivation: averaging filters can lead to excessive dissipation in the barotropic mode

Objective: put the minimum amount of dissipation to stabilize the splitting

Diffusion is introduced within the barotropic time-stepping rather than averaging filters by adapting the parameters in the generalized forward-backward scheme

$$(\beta, \gamma, \varepsilon) = (0.281105, 0.08344500 - 0.51358400\alpha_d, 0.00976186 - 0.13451357\alpha_d)$$

with $\alpha_d \approx 0.5$.

Remarks:

- This option may require to increase $\text{NDTFAST} = \Delta t_{3D} / \Delta t_{2D}$ because the stability constraint of the modified generalized forward-backward scheme is less than the one of the original generalized forward-backward scheme.
- The filter_none approach is systematically more efficient than averaging filters

7.2.5 Stability constraints

- **Barotropic mode** (note that considering an Arakawa C-grid divides the theoretical stability limit by a factor of 2)

$$\Delta t \sqrt{gH \left(\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2} \right)} \leq 0.89$$

- **3D advection**

$$\alpha_{\text{adv}}^x + \alpha_{\text{adv}}^y + \beta \alpha_{\text{adv}}^z \leq \alpha_{\text{horiz}}^*$$

where α_{adv}^x , α_{adv}^y , and α_{adv}^z are the Courant numbers in each direction and $\beta = \alpha_{horiz}^*/\alpha_{vert}^*$ a coefficient arising from the fact that different advection schemes with different stability criteria may be used in the horizontal and vertical directions. Typical CFL values for α_{horiz}^* and α_{vert}^* with Croco time-stepping algorithm are

Advection scheme	Max Courant number (α^*)
C2	1.587
UP3	0.871
SPLINES	0.916
C4	1.15
UP5	0.89
C6	1.00

- **Internal waves**

$$\Delta t c_1 \sqrt{\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2}} \leq 0.843686$$

where c_1 the phase speed associated with the first (fastest) baroclinic mode.

- **Coriolis**

$$f \Delta t \leq 1.58$$

7.3 Advection Schemes

7.3.1 Lateral Momentum Advection

Related CPP options:

UV_HADV_UP3	Activate 3rd-order upstream biased advection scheme
UV_HADV_UP5	Activate 5th-order upstream biased advection scheme
UV_HADV_C2	Activate 2nd-order centred advection scheme (should be used with explicit momentum mixing)
UV_HADV_C4	Activate 4th-order centred advection scheme (should be used with explicit momentum mixing)
UV_HADV_C6	Activate 6th-order centred advection scheme (should be used with explicit momentum mixing)
UV_HADV_WENO5	Activate WENO 5th-order advection scheme
UV_HADV_TVD	Activate Total Variation Diminishing scheme

Preselected options:

```
# define UV_HADV_UP3
# undef UV_HADV_UP5
# undef UV_HADV_C2
# undef UV_HADV_C4
# undef UV_HADV_C6
# undef UV_HADV_WENO5
# undef UV_HADV_TVD
```

These options are set in `set_global_definitions.h` as the default `UV_HADV_UP3` is the only one recommended for standard users.

7.3.2 Lateral Tracer advection

Related CPP options:

<code>TS_HADV_UP3</code>	3rd-order upstream biased advection scheme
<code>TS_HADV_RSUP3</code>	Split and rotated 3rd-order upstream biased advection scheme
<code>TS_HADV_UP5</code>	5th-order upstream biased advection scheme
<code>TS_HADV_RSUP5</code>	Split and rotated 5th-order upstream biased advection scheme with reduced dispersion/diffusion
<code>TS_HADV_C4</code>	4th-order centred advection scheme
<code>TS_HADV_C6</code>	Activate 6th-order centred advection scheme
<code>TS_HADV_WENO5</code>	5th-order WENOZ quasi-monotonic advection scheme for all tracers
<code>BIO_HADV_WENO5</code>	5th-order WENOZ quasi-monotone advection scheme for passive tracers (including biology and sediment tracers)

Preselected options:

```
# undef TS_HADV_UP3
# define TS_HADV_RSUP3
# undef TS_HADV_UP5
# undef TS_HADV_RSUP5
# undef TS_HADV_C4
# undef TS_HADV_C6
# undef TS_HADV_WENO5
# if defined PASSIVE_TRACER || defined BIOLOGY || defined SEDIMENT
# define BIO_HADV_WENO5
# endif
```

`TS_HADV_RSUP3` is recommended for realistic applications with variable bottom topography as it strongly reduces diapycnal mixing. It splits the `UP3` scheme into 4th-order centered advection and rotated bilaplacian diffusion with grid-dependent diffusivity. It calls for CPP options in `set_global_definitions.h` for the explicit treatment of bilaplacian diffusion (see below). `TS_HADV_RSUP3` is expensive in terms of computational cost and requires more than 30 sigma levels to perform properly. Therefore, for small domains dominated by open boundary fluxes, `TS_HADV_UP5` may present a cheaper alternative and good compromise. `TS_HADV_RSUP5` is still experimental but allows a decrease in numerical diffusivity compared to `TS_HADV_RSUP3` by using 6th order rather than 4th-order centered advection (it resembles in spirit a split-rotated `UP5` scheme but the use of bilaplacian rather than trilaplacian diffusion keeps it 3rd order). `TS_HADV_C4` has no implicit diffusion and is thus accompanied by rotated Smagorinsky diffusion defined in `set_global_definitions.h`; it is not recommended for usual applications. For `RSUP` family, by default the diffusive part is oriented along geopotential.

7.3.3 Vertical Momentum advection

Related CPP options:

UV_VADV_SPLINES	4th-order compact advection scheme
UV_VADV_C2	2nd-order centered advection scheme
UV_VADV_WENO5	5th-order WENOZ quasi-monotone advection scheme
UV_VADV_TVD	Total Variation Diminishing (TVD) scheme

Preselected options:

```
#ifndef UV_VADV_SPLINES
#elif defined UV_VADV_WENO5
#elif defined UV_VADV_C2
#elif defined UV_VADV_TVD
#else
# define UV_VADV_SPLINES
# undef UV_VADV_WENO5
# undef UV_VADV_C2
# undef UV_VADV_TVD
#endif
```

7.3.4 Vertical Tracer advection

Related CPP options:

TS_VADV_SPLINES	4th-order compact advection scheme
TS_VADV_AKIMA	4th-order centered advection scheme with harmonic averaging
TS_VADV_C2	2nd-order centered advection scheme
TS_VADV_WENO5	5th-order WENOZ quasi-monotone advection scheme

Preselected options:

```
#ifndef TS_VADV_SPLINES
#elif defined TS_VADV_AKIMA
#elif defined TS_VADV_WENO5
#elif defined TS_VADV_C2
#else
# undef TS_VADV_SPLINES
# define TS_VADV_AKIMA
# undef TS_VADV_WENO5
# undef TS_VADV_C2
#endif
```

7.3.5 Adaptively implicit vertical advection

Related CPP options:

VADV_ADAPT_IMP	Activate adaptive, Courant number dependent implicit advection scheme
VADV_ADAPT_PRED	Adaptive treatment at both predictor and corrector steps

Preselected options:

```

#ifdef VADV_ADAPT_IMP
# undef VADV_ADAPT_PRED
# define UV_VADV_SPLINES
# undef UV_VADV_C2
# undef UV_VADV_WENO5
# undef UV_VADV_TVD
#endif
#ifdef VADV_ADAPT_IMP
# define TS_VADV_SPLINES
# undef TS_VADV_AKIMA
# undef TS_VADV_WENO5
# undef TS_VADV_C2
#endif
    
```

7.3.6 Numerical details on advection schemes

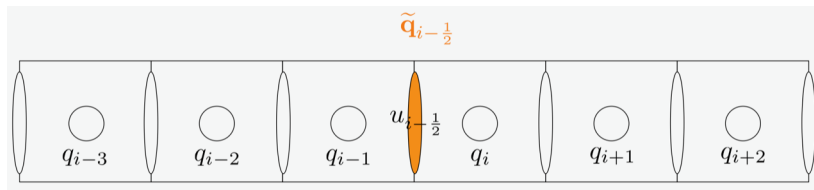


Fig. 2: Fig: variable location on an Arakawa C-grid. Tracer values are cell centered while velocities are defined on interfaces.

$$\partial_x(uq)|_{x=x_i} = \frac{1}{\Delta x_i} \left\{ u_{i+\frac{1}{2}} \tilde{q}_{i+\frac{1}{2}} - u_{i-\frac{1}{2}} \tilde{q}_{i-\frac{1}{2}} \right\}$$

Linear advection schemes

$$\tilde{q}_{i-\frac{1}{2}}^{\text{C2}} = \frac{q_i + q_{i-1}}{2} \quad (7.1)$$

$$\tilde{q}_{i-\frac{1}{2}}^{\text{C4}} = \left(\frac{7}{6}\right) \tilde{q}_{i-\frac{1}{2}}^{\text{C2}} - \left(\frac{1}{12}\right) (q_{i+1} + q_{i-2}) \quad (7.2)$$

$$\tilde{q}_{i-\frac{1}{2}}^{\text{UP3}} = \tilde{q}_{i-\frac{1}{2}}^{\text{C4}} + \text{sign}\left(\frac{1}{12}, u_{i-\frac{1}{2}}\right) (q_{i+1} - 3q_i + 3q_{i-1} - q_{i-2}) \quad (7.3)$$

$$\tilde{q}_{i-\frac{1}{2}}^{\text{C6}} = \left(\frac{8}{5}\right) \tilde{q}_{i-\frac{1}{2}}^{\text{C4}} - \left(\frac{19}{60}\right) \tilde{q}_{i-\frac{1}{2}}^{\text{C2}} + \left(\frac{1}{60}\right) (q_{i+2} + q_{i-3}) \quad (7.4)$$

$$\tilde{q}_{i-\frac{1}{2}}^{\text{UP5}} = \tilde{q}_{i-\frac{1}{2}}^{\text{C6}} - \text{sign}\left(\frac{1}{60}, u_{i-\frac{1}{2}}\right) (q_{i+2} - 5q_{i+1} + 10q_i - 10q_{i-1} + 5q_{i-2} - q_{i-3}) \quad (7.5)$$

Split upwind schemes

Because odd-ordered advection schemes can be formulated as the sum of the next higher-order (centered) advection scheme with a dissipation term it is possible to split the purely centered and dissipative parts of UP3 and UP5 schemes. In this case the centered part is treated within the predictor-corrector framework while the flow-dependent dissipative part is treated with a one-step Euler scheme. Such splitting has two advantages:

1. It allows better stability for SUP3 and SUP5 schemes compared to UP3 and UP5 schemes.
2. Isolating the dissipative part allows to rotate it in the neutral direction to reduce spurious diapycnal mixing (RSUP3 scheme).

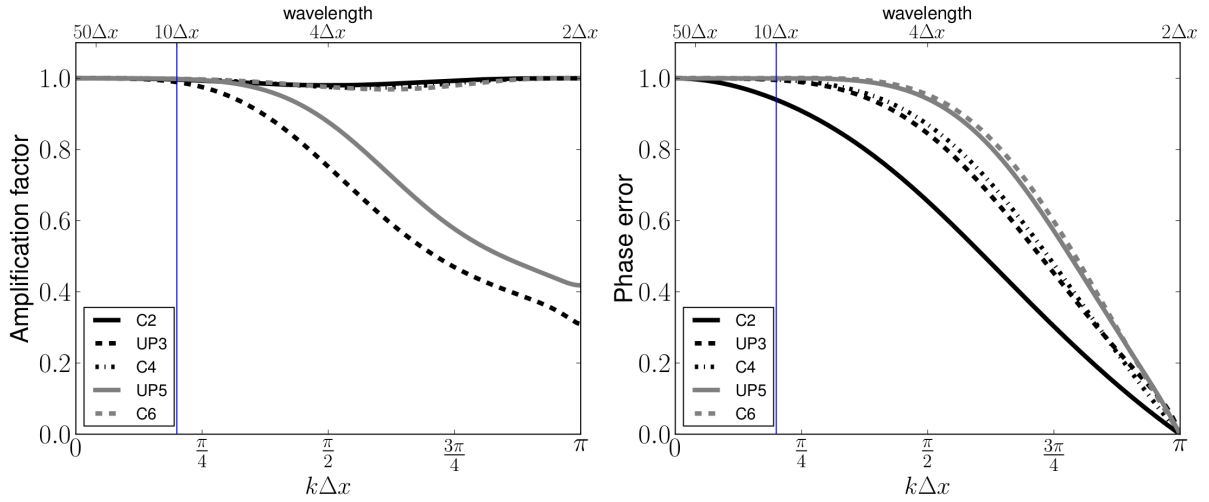


Fig. 3: Fig: amplification errors (left) and phase errors (right) for linear advection of order 2 to 6.

Splines reconstruction and Akima 4th-order schemes

Similar to a 4th-order compact scheme, the interfacial values for the splines reconstruction scheme are obtained as a solution of a tridiagonal problem

$$\text{Hz}_{k+1}\tilde{q}_{k-\frac{1}{2}} + 2(\text{Hz}_k + \text{Hz}_{k+1})\tilde{q}_{k+\frac{1}{2}} + \text{Hz}_k\tilde{q}_{k+\frac{3}{2}} = 3(\text{Hz}_k\bar{q}_{k+1} + \text{Hz}_{k+1}\bar{q}_k)$$

where \bar{q}_k values should be understood in a finite-volume sense (i.e. as an average over a control volume).

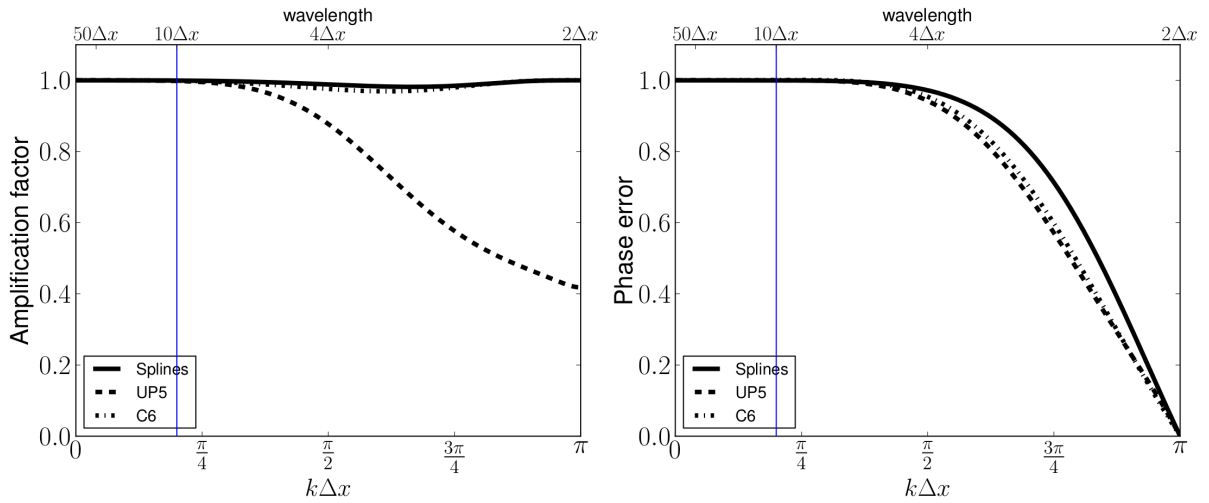


Fig. 4: Fig: amplification errors (left) and phase errors (right) for linear advection of order 5 and 6 and for Splines reconstruction.

The AKIMA scheme corresponds to a 4th-order accurate scheme where an harmonic averaging of the slopes is used instead of the algebraic average used for a standard C4 scheme

$$\tilde{q}_{k+\frac{1}{2}} = \frac{q_{k+1} + q_k}{2} - \frac{\bar{\delta}q_{k+1} - \bar{\delta}q_k}{6} \quad \bar{\delta}q_k = \begin{cases} 2 \frac{\delta q_{k+\frac{1}{2}} \delta q_{k-\frac{1}{2}}}{\delta q_{k+\frac{1}{2}} + \delta q_{k-\frac{1}{2}}}, & \text{if } \delta q_{k+\frac{1}{2}} \delta q_{k-\frac{1}{2}} > 0 \\ 0, & \text{otherwise} \end{cases}$$

Adaptively implicit vertical advection

Idea: the vertical velocity Ω is split between an explicit and implicit contribution depending on the local Courant number

$$\Omega = \Omega^{(e)} + \Omega^{(i)}, \quad \Omega^{(e)} = \frac{\Omega}{f(\alpha_{\text{adv}}^z, \alpha_{\text{max}})}, \quad f(\alpha_{\text{adv}}^z, \alpha_{\text{max}}) = \begin{cases} 1, & \alpha_{\text{adv}}^z \leq \alpha_{\text{max}} \\ \alpha/\alpha_{\text{max}}, & \alpha_{\text{adv}}^z > \alpha_{\text{max}} \end{cases}$$

- $\Omega^{(e)}$ is integrated with an explicit scheme with CFL α_{max} .
- $\Omega^{(i)}$ is integrated with an implicit upwind Euler scheme.
- $f(\alpha_{\text{adv}}^z, \alpha_{\text{max}})$ is a function responsible for the splitting of Ω between an explicit and an implicit part.

This approach has the advantage to render vertical advection unconditionally stable and to maintain good accuracy in locations with small Courant numbers. The current implementation is based on the SPLINES scheme for the explicit part.

Total variation bounded scheme (WENO5)

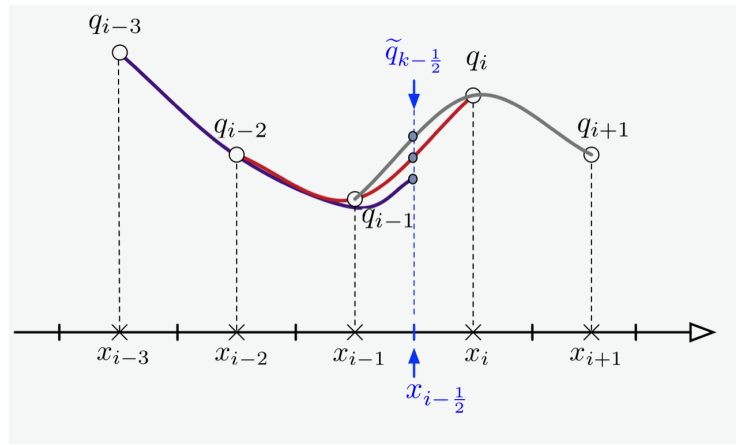


Fig. 5: Fig: different stencils used to evaluate the interfacial value $\tilde{q}_{k+\frac{1}{2}}$ with WENO5 scheme

Nonlinear weighting between 3 evaluations of interfacial values based on 3 different stencils

$$\tilde{q}_{k-\frac{1}{2}} = w_0 \tilde{q}_{k-\frac{1}{2}}^{(0)} + w_1 \tilde{q}_{k-\frac{1}{2}}^{(1)} + w_2 \tilde{q}_{k-\frac{1}{2}}^{(2)}$$

where the weights are subject to the following constraints:

1. Convexity $\sum_{j=0}^2 w_j = 1$.
2. ENO property (Essentially non oscillatory).
3. 5th-order if $q(x)$ is smooth.

The resulting scheme is not monotonicity-preserving but instead it is Total Variation Bounded (TVB).

Total variation diminishing scheme

Upwinding of nonlinear terms

In CROCO the nonlinear advection terms are formulated as in Lilly (1965) :

$$\partial_t(\text{Hz}u) + \partial_x((\text{Hz}u)u) + \partial_y((\text{Hz}v)u) + \dots \quad (7.6)$$

$$\partial_t(\text{Hz}v) + \partial_x((\text{Hz}u)v) + \partial_y((\text{Hz}v)v) + \dots \quad (7.7)$$

which are discretised with third order accuracy as

$$\left(\widetilde{(\text{Hz } u)u} \right)_{i,j} = \left(\widetilde{\text{Hz } u} \right)_{i,j}^{C4} \widetilde{u}_{i,j}^{\text{UP3}} \quad (7.8)$$

$$\left(\widetilde{(\text{Hz } v)u} \right)_{i+\frac{1}{2},j+\frac{1}{2}} = \left(\widetilde{\text{Hz } v} \right)_{i+\frac{1}{2},j+\frac{1}{2}}^{C4} \widetilde{u}_{i+\frac{1}{2},j+\frac{1}{2}}^{\text{UP3}} \quad (7.9)$$

where the direction for upwinding is selected considering

$$u_{i,j}^{\text{upw}} = u_{i+\frac{1}{2},j} + u_{i-\frac{1}{2},j}, \quad v_{i+\frac{1}{2},j+\frac{1}{2}}^{\text{upw}} = (\text{Hz } v)_{i,j+\frac{1}{2}} + (\text{Hz } v)_{i+1,j+\frac{1}{2}}$$

7.4 Pressure gradient

7.5 Equation of State

Related CPP options:

SALINITY	Activate salinity as an active tracer
NONLIN_EOS	Activate nonlinear equation of state
SPLIT_EOS	Activate the split of the nonlinear equation of state in adiabatic and compressible parts for reduction of pressure gradient errors

Preselected options:

```
# define SALINITY
# define NONLIN_EOS
# define SPLIT_EOS
```

7.6 Wetting and Drying

The processes of wetting and drying have important physical and biological impacts on shallow water systems. Flooding and dewatering effects on coastal mud flats and beaches occur on various time scales ranging from storm surge, periodic rise and fall of the tide, to infragravity wave motions. To correctly simulate these physical processes with a numerical model requires the capability of the computational cells to become flooded and dewatered. Warner et al. (2013) proposed a method for wetting and drying based on an approach consistent with a cell-face blocking algorithm. The method allows water to always flow into any cell, but prevents outflow from a cell when the total depth in that cell is less than a user defined critical value. See Warner et al. (2013) for details.

The Wetting-Drying scheme is derived from John Warner's code (Rutgers ROMS) and adapted to the time stepping scheme of CROCO. The main idea is to cancel the outgoing momentum flux (not the incoming) from a grid cell if its total depth is below a threshold value (critical depth D_{crit} between 5 and 20 cm according to local slope; D_{crit} min and max adjustable in param.h). This scheme is tested in the Thacker case producing oscillations in a rotating bowl for which an analytical solution is known.

7.7 Non-Boussinesq Solver

CROCO can be used in a Boussinesq hystrostatic mode, or a non-hydrostatic, non-boussinesq mode (NBQ). The Non-Hydrostatic approach is based on the relaxation of the Boussinesq approximation instead of solving a Poisson system. It replaces the barotropic mode solver by a fully 3D fast mode solver, resolving all waves down to acoustic waves. The barotropic mode is part of the fast mode in this case. Depending on the physical problem, the sound speed can be decreased to the maximum wave velocity one wants to solve. The NH solver can be used in problems from a few tens of meters to LES or DNS resolutions. It comes with monotonicity preserving advection schemes (WENO5, TVD) and fully 3D turbulent closure schemes.

Related CPP options (for users):

NBQ	Activates Non-hysrostatic, non-Boussinesq solver
-----	--

PARAMETRIZATIONS

8.1 Vertical mixing parametrizations

CROCO contains a variety of methods for setting the vertical viscous and diffusive coefficients. The choices range from simply choosing fixed values to the KPP and the generic lengthscale (GLS) turbulence closure schemes. See Large (1998) for a review of surface ocean mixing schemes. Many schemes have a background molecular value which is used when the turbulent processes are assumed to be small (such as in the interior).

Related CPP options:

ANA_VMIX	Analytical definition
BVF_MIXING	Brunt-Vaisala frequency based
LMD_MIXING	K-profile parametrisation
GLS_MIXING	Generic lengthscale parametrisation

Preselected options:

```
NONE : default is no mixing scheme
```

8.1.1 Analytical definition

Related CPP options:

ANA_VMIX	Analytical definition
----------	-----------------------

Preselected options:

```
NONE
```

A profile for mixing coefficient $K_{m,s}(z)$ can be set in ana_vmix routine for variables Akv (viscosity) and Akt (diffusivity), which is called at each time step. In this case, background coefficients read in croco.in can be used.

8.1.2 BVF mixing

Related CPP options:

BVF_MIXING	Brunt-Vaisala frequency based
------------	-------------------------------

Preselected options:

```
NONE
```

It computes diffusivity using a Brunt-Vaisala frequency based vertical mixing scheme. Viscosity is set to its background. In static unstable regime, diffusivity is enhanced.

- If $N^2(z) < 0$:

$$K_{m,s}(z) = 0.1 \text{ m}^2 \text{ s}^{-1}$$

- If $N^2(z) > 0$:

$$K_{m,s}(z) = 10^{-7} / \sqrt{N^2(z)}, \quad K_{m,s}^{\min} \leq K_{m,s}(z) \leq K_{m,s}^{\max}$$

Default bounds are quite restrictive :

$$K_{m,s}^{\min} = 3 \times 10^{-5} \text{ m}^2 \text{ s}^{-1}, \quad K_{m,s}^{\max} = 4 \times 10^{-4} \text{ m}^2 \text{ s}^{-1}$$

8.1.3 K-profile parametrization

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

Related CPP options:

LMD_MIXING	K-profile parametrisation
LMD_SKPP	
LMD_SKPP2005	
LMD_BKPP	
LMD_RIMIX	
LMD_CONVEC	
LMD_BKPP	
LMD_BKPP2005	

Preselected options:

```
#if defined LMD_SKPP
# define LMD_SKPP2005
#endif
#ifndef LMD_BKPP
# undef LMD_BKPP2005
#endif
```

Surface boundary layer

- LMD_SKPP (Large et al, 1994)
 - Step 1 : Compute boundary layer depth $h_{bl}(z_r \rightarrow z_N)$

$$\text{Ri}_b(z) = \frac{g(z_r - z) (\rho(z) - \rho_r) / \rho_0}{|\mathbf{u}(z) - (\mathbf{u}_h)_r|^2 + V_t^2(z)}, \quad \text{Ri}_b(-h_{bl}) = \text{Ri}_{cr}$$

- Step 2 : In the stable case $\text{math}::(B_f > 0) : h_{bl} = \min(h_{bl}, h_{ek}, h_{mo})$

$$h_{ek} = 0.7u_* / f, \quad h_{mo} = u_*^3 / (\kappa B_f).$$

- step 3 : Compute turbulent viscosity and diffusivity

$$K_{m,s}(z) = w_{m,s} h_{bl} G(z/h_{bl}), \quad w_{m,s} = \kappa u_* \psi_{m,s}(zB_f/u_*^3)$$

Choice of the critical Richardson number Ri_{cr} : $Ri_{cr} \in [0.15, 0.45]$

- LMD_SKPP2005 (Shchepetkin et al, 2005)

- Criteria for h_{bl} : integral layer where production of turbulence by shear balances dissipation by the stratification

$$Cr(z) = \int_z^\zeta \mathcal{K}(z') \left\{ |\partial_{z'} \mathbf{u}_h|^2 - \frac{N^2}{Ri_{cr}} - C_{Ek} f^2 \right\} dz' + \frac{V_t^2(z)}{(\zeta - z)}, \quad Cr(-h_{bl}) = 0$$

- Consistent with the original KPP

$$Cr(-h_{bl}) = 0 \Rightarrow \frac{(\zeta - z) \int_z^\zeta \mathcal{K}(z') N^2(z') dz'}{(\zeta - z) \int_z^\zeta \mathcal{K}(z') \left\{ |\partial_{z'} \mathbf{u}_h|^2 - C_{Ek} f^2 \right\} dz' + V_t^2(z)} = Ri_{cr}$$

Advantages :

-> consistent with Ekman problem

-> tends to give deeper boundary layers : $(\zeta - z) \int_z^\zeta |\partial_{z'} \mathbf{u}_h|^2 dz' \geq |\mathbf{u}_h(z) - \mathbf{u}_h(\zeta)|^2$.

Interior scheme

$$K_{m,s}(z) = K_{m,s}^{sh}(z) + K_{m,s}^{iw}(z) + K_{m,s}^{dd}(z)$$

- cpp key LMD_RIMIX, RI_(H-V)SMOOTH (Large et al., 1994)

$$Ri_g = N^2 / [(\partial_z u)^2 + (\partial_z v)^2]$$

$$K_{m,s}^{sh}(z) = \begin{cases} K_{0,c} & Ri_g < 0 \quad \leftarrow \text{[LMD_CONVEC]} \\ K_0 \left[1 - \left(\frac{Ri_g}{Ri_0} \right)^3 \right] & 0 < Ri_g < Ri_0 \\ 0 & Ri_0 < Ri_g \end{cases}$$

$$K_0 = 5 \times 10^{-3} \text{ m}^2 \text{ s}^{-1}, Ri_0 = 0.7$$

- cpp key LMD_NUW_GARGETT (Gargett & Holloway)

$$K_m^{iw}(z) = \frac{10^{-6}}{\sqrt{\max(N^2(z), 10^{-7})}}, \quad K_s^{iw}(z) = \frac{10^{-7}}{\sqrt{\max(N^2(z), 10^{-7})}}$$

- cpp key LMD_DDMIX (cf Large et al., 1994, eqns (31))

Bottom boundary layer

- cpp key LMD_BOTEK : Bottom Ekman layer

$$\begin{aligned}
 h_{\text{Ek}} &= \min \left\{ \frac{0.3u_{*,b}}{|f|}, h \right\} \\
 \sigma_{k+\frac{1}{2}} &= (z_{k+\frac{1}{2}} - h)/h_{\text{Ek}} \\
 K_{k+\frac{1}{2}}^{\text{Ek}} &= \max \{ 4 \kappa u_{*,b} h_{\text{Ek}} \sigma(1 - \sigma), K_{\min} \} \\
 \text{AKv}_{k+\frac{1}{2}} &= \text{AKv}_{k+\frac{1}{2}} + K_{k+\frac{1}{2}}^{\text{Ek}} \\
 \text{AKt}_{k+\frac{1}{2}} &= \text{AKt}_{k+\frac{1}{2}} + K_{k+\frac{1}{2}}^{\text{Ek}}
 \end{aligned}$$

- cpp key LMD_BKPP (Bottom KPP 1994)

Same rationale than surface KPP but this time we search for the critical value $\text{Ri}_{\text{cr}} (\approx 0.3)$ starting from the bottom

$$h_{\text{bbl}} = \min \left(h_{\text{bbl}}, \frac{0.7u_{*,b}}{|f|} \right) K_{m,s}(z) = \kappa u_{*,b} h_{\text{bbl}} G(\sigma), \quad \sigma = \frac{(z - h)}{h_{\text{bbl}}}$$

8.1.4 Generic length scale

Related CPP options:

GLS_MIXING	Activate OpenMP parallelization protocol
GLS_KEPSILON	
GLS_KOMEGA	Activate MPI parallelization protocol
CANUTO_A	Compute the best decomposition for OpenMP
CANUTO_B	
MelYam_82	Output one file per CPU
KanCla_94	Use NetCDF4 capabilities
Luyten_96	Dedicated CPU for output (needs XIOS installed)
Cheng_02	

Preselected options:

```

#ifdef GLS_MIXING
# if defined GLS_KOMEGA
# elif defined GLS_KEPSILON
# elif defined GLS_GEN
# else
# define GLS_KEPSILON
# endif
# if defined CANUTO_A
# elif defined GibLau_78
# elif defined MelYam_82
# elif defined KanCla_94
# elif defined Luyten_96
# elif defined CANUTO_B
# elif defined Cheng_02
# else
# define CANUTO_A
# endif
#endif
    
```

Umlauf and Burchard (2003) have come up with a generic two-equation turbulence closure scheme which can be tuned to behave like several of the traditional schemes, including that of Mellor and Yamada 2.5 (above). This is known as the Generic Length Scale, or GLS vertical mixing scheme and was introduced to ROMS in Warner et al. (2005).

The generic equations are formulated as :

$$\begin{aligned}\partial_t e &= \partial_z (K_e \partial_z e) + P + B - \varepsilon, & K_e &= K_M / \text{Sc}_e \\ \partial_t \psi &= \partial_z (K_\psi \partial_z \psi) + \psi e^{-1} (\beta_1 P + \beta_3^\pm B - \beta_2 \varepsilon), & K_\psi &= K_M / \text{Sc}_\psi\end{aligned}$$

with

$$psi = (c_\mu^0)^p e^{ml^n}, \quad P = K_m [(\partial_z u)^2 + (\partial_z v)^2], \quad B = -K_s N^2$$

and

$$\varepsilon = (c_\mu^0)^{3+\frac{p}{n}} e^{\frac{3}{2}+\frac{m}{n}} \psi^{-\frac{1}{n}}, \quad l = (c_\mu^0)^3 e^{\frac{3}{2}} \varepsilon^{-1}$$

We end up with :

$$K_m = c_\mu \left(\frac{e^2}{\varepsilon} \right), \quad K_s = c_\mu' \left(\frac{e^2}{\varepsilon} \right)$$

A GLS instance is characterized by :

- a choice for the exponents m, n and p (e.g. $k - \varepsilon, k - \omega, k - kl, \text{gen}'$)
- a value of the coefficients $\beta_1; \beta_2$ and β_3^\pm
- a value of Schmidt number Sc_e and Sc_ψ

Computation of c_μ, c_μ' :

$$\begin{aligned}c_\mu &= \frac{n_0 + n_1 \alpha_N + n_2 \alpha_M}{d_0 + d_1 \alpha_N + d_2 \alpha_M + d_3 \alpha_N \alpha_M + d_4 \alpha_N^2 + d_5 \alpha_M^2} \\ c_\mu' &= \frac{n_0' + n_1' \alpha_N + n_2' \alpha_M}{d_0 + d_1 \alpha_N + d_2 \alpha_M + d_3 \alpha_N \alpha_M + d_4 \alpha_N^2 + d_5 \alpha_M^2}\end{aligned}$$

with

$$\alpha_N = -N^2 \left(\frac{e}{\varepsilon} \right)^2, \quad \alpha_M = [(\partial_z u)^2 + (\partial_z v)^2] \left(\frac{e}{\varepsilon} \right)^2$$

The parameters d_j, n_j and n_j' are different for each stability function :

- Canuto et al., 2001(A-B);
- Gibson & Launder, 1978;
- Mellor & Yamada, 1982;
- Kantha & Clayson, 1994;
- Luyten, 1996;
- Cheng et al., 2002; \$hdots\$

8.2 Bottom friction

Related CPP options:

LIMIT_BSTRESS	Bottom stress limitation for stability
BSTRESS_FAST	Bottom stress computed in step3d_fast
BBL	Bottom boundary layer parametrization

Specification in croco.in:

bottom_drag:	RDRG [m/s],	RDRG2,	Zob [m],	Cdb_min,	Cdb_max
	3.0d-04	0.d-3	0.d-3	1.d-4	1.d-1

- General form for 3D equations (cf get_vbc.F) :

– If $z_{0,b} \neq 0 \rightarrow$ quadratic friction with log-layer ($C_{d,\min} \leq C_d \leq C_{d,\max}$)

$$\boldsymbol{\tau}_b = C_d \|\mathbf{u}_{k=1}\| \mathbf{u}_{k=1}, \quad C_d = \left(\frac{\kappa}{\ln([z_1 - H]/z_{0,b})} \right)^2$$

– If $r_{\text{drg2}} > 0 \rightarrow$ quadratic friction with $C_d = \text{constant}$ }

$$\boldsymbol{\tau}_b = r_{\text{drg2}} \|\mathbf{u}_{k=1}\| \mathbf{u}_{k=1},$$

– Otherwise \rightarrow linear friction

$$\boldsymbol{\tau}_b = r_{\text{drg}} \mathbf{u}_{k=1},$$

- In the barotropic mode (cf step2D.F) :

$$\boldsymbol{\tau}_b^{2d} = (r_{\text{drg}} + r_{\text{drg2}} \|\bar{\mathbf{u}}\|) \bar{\mathbf{u}}$$

to be continued here for BSTRESS_FAST and BBL ...

PARALLELISATION

CROCO has been designed to be optimized on both shared and distributed memory parallel computer architectures. Parallelization is done by two dimensional sub-domains partitioning. Multiple sub-domains can be assigned to each processor in order to optimize the use of processor cache memory. This allow super-linear scaling when performance growth even faster than the number of CPUs.

Related CPP options:

OPENMP	Activate OpenMP parallelization protocol
MPI	Activate MPI parallelization protocol
MPI_NOLAND	No computation on land only CPUs (needs preprocessing)
AUTO_TILING	Compute the best decomposition for OpenMP
PARALLEL_FILES	Output one file per CPU
NC4_PAR	Use NetCDF4 capabilities
XIOS	Dedicated CPU for output (needs XIOS installed)

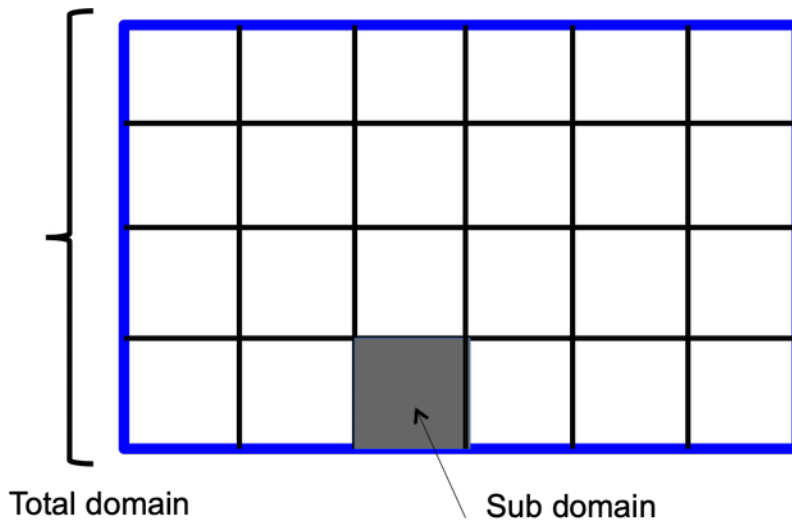
Preselected options:

```
# undef MPI
# undef OPENMP
# undef MPI_NOLAND
# undef AUTOTILING
# undef PARALLEL_FILES
# undef NC4_PAR
# undef XIOS
```

9.1 Parallel strategy overview

Two kind of parallelism are currently supported by CROCO : MPI (distributed memory) and OpenMP (shared memory). COROC doesn't currently support hybrid parallelisation : use of cpp keys MPI or OPENMP is exclusive.

9.1.1 OpenMP (#define OPENMP)



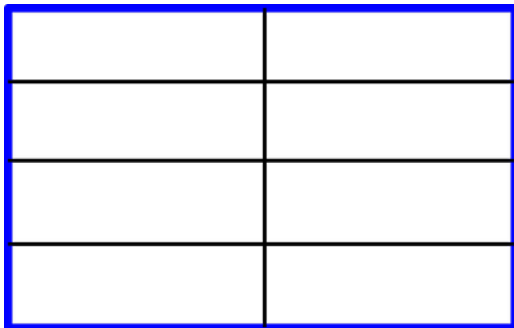
Variables in param.h:

- NPP : number of threads
- NSUB_X : number of tiles in XI direction
- NSUB_E : number of threads in ETA direction

NSUB_X x NSUB_E has to be a multiple of NPP. Most of the time, we set $NPP = NSUB_X \times NSUB_E$

Example 1:

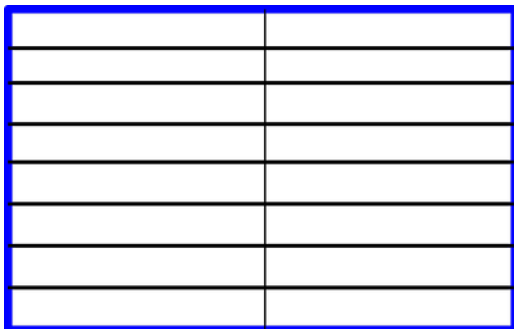
One node with 8 cores: NPP=8, NSUB_X=2, NSUB_ETA=4



Each thread computes **one** sub-domain.

Example 2:

Still one node with 8 cores: NPP=8, NSUB_X=2, NSUB_E=8



Each thread computes **two** sub-domains.

Code structure

- OpenMP is **NOT** implemented at loop level
- but uses a domain decomposition (similar to MPI) with parallel region
- use of *First touch initialisation* so working arrays are attached to the same thread
- working arrays have the size of the sub-domain only

```
C$OMP PARALLEL
  Call step3D_t_thread()
C$OMP END PARALLEL
```

Fig. 1: Example of a parallel region

```
Do tile=my_first,my_last ! Loop on the tiles computed
                        ! by the current thread
  Call compute_1(tile) ! No Synchronisation needed
  Call compute_2(tile) ! by these procedures
Enddo
C$OMP BARRIER ! synchronisation

Do tile=my_first,my_last ! Loop on the tiles computed
                        ! by the current thread
  Call compute_3(tile) ! No Synchronisation needed
  Call compute_4(tile) !
Enddo
C$OMP BARRIER ! synchronisation
```

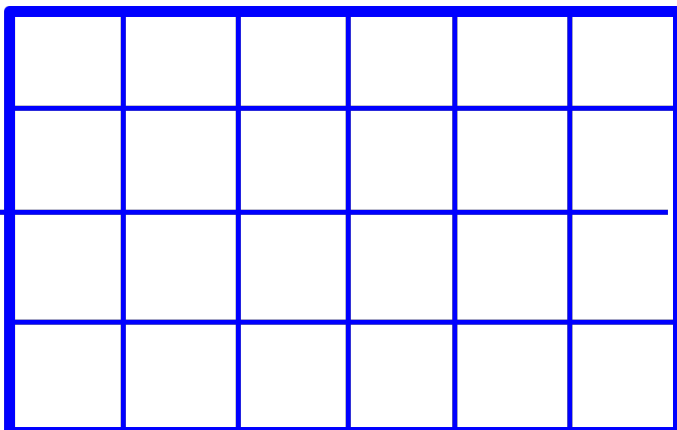
Fig. 2: Inside a parallel region

Here Compute_1 and Compute2 can't write on the same index of a global array.

9.1.2 MPI (#define MPI)

Variables in param.h:

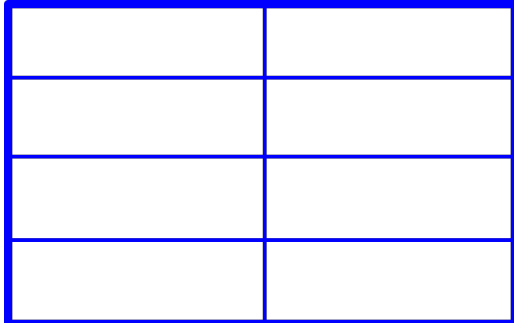
- NP_XI : decomposition in XI direction
- NP_ETA : decomposition in ETA direction
- NNODES : number of cores (=NP_XI x NP_ETA, except with MPI_NOLAND)
- NPP = 1
- NSUB_X and NSUB_ETA, number of sub-tiles (almost always =1)



Example 1:

8 cores:

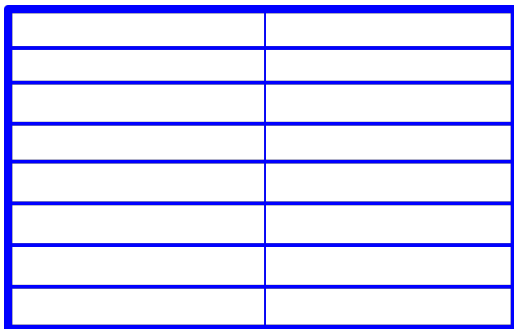
- NP_XI=2, NP_ETA=4, NNODES=8
- NPP=1, NSUB_X=1, NSUB_ETA=1



Example 2:

8 cores:

- NP_XI=2, NP_ETA=4, NNODES=8
- NPP=1, NSUB_X=1, NSUB_ETA=2

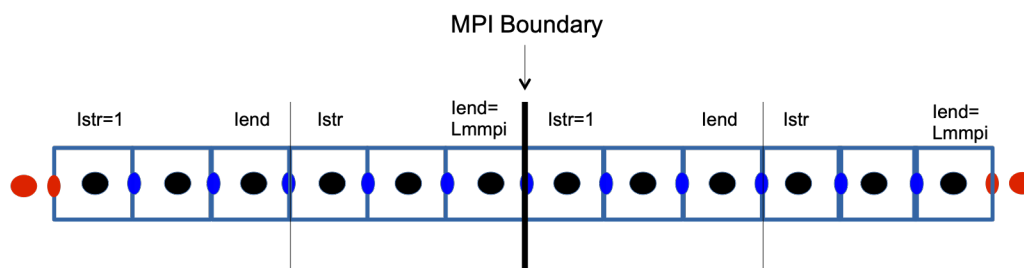


9.2 Loops and indexes

Parallel/sequential correspondance:

Decomposition:

Example : 2 MPI domains, with 2 sub-domains (OpenMP or not) by domain MPI



Istr, lend are the limits of the sub-domains (without overlap). There are calculated dynamically.

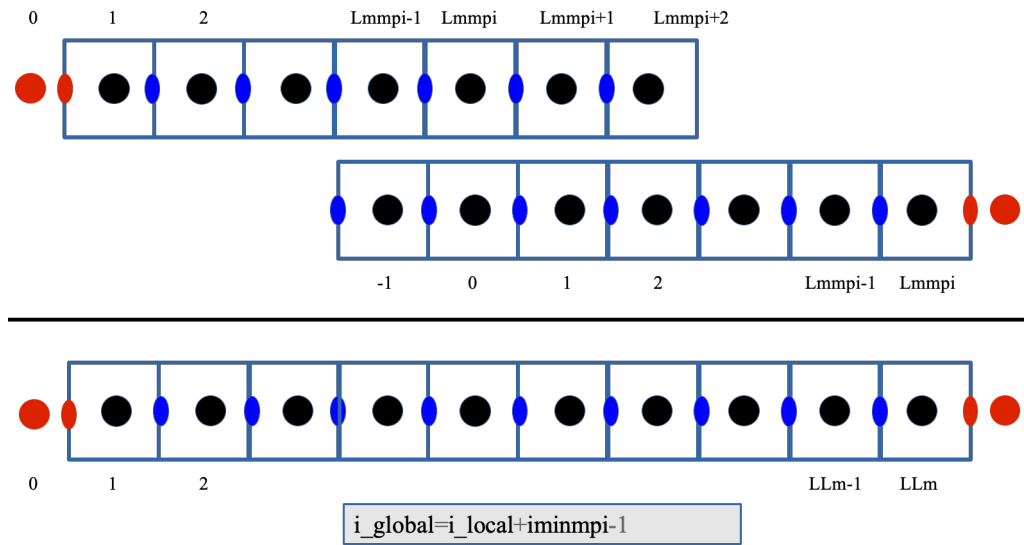


Fig. 3: Decomposition on 2 sub-domain (up), total (sequential) domain (bottom)

```

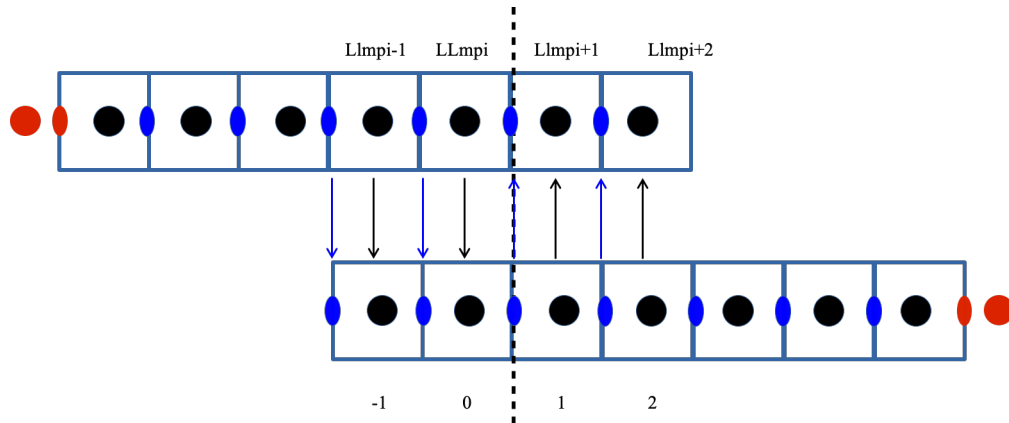
Subroutine compute_1(tile)
Integer tile, trd
#include « private_scratch.h » ! Private work arrays
! A2d, A3d
#include « compute_tile_bounds.h » Compute Istr, Iend

trd=0
CS  trd=omp_get_thread_num()
Call compute_1_tile(Istr,Iend,Jstr,Jend,A2d(1,1,trd),A3d(1,1,trd))
Return
end
    
```

Fig. 4: Computation of Istr, Iend and use of working arrays

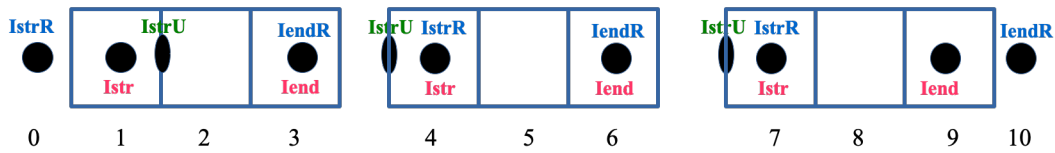
9.3 Exchanges

CROCO makes use 2 or 3 ghost cells depending on the numerical schemes chosen.



In the example above (2 ghosts cells), for correct exchanges, after computation:

- η has to be valid on (1:Iend)
- u has to be valid on (1:Iend) except on the left domain (2:Iend)



IstrU is the limit of validity at U point

```

Subroutine compute_1_tile(Istr, Iend ...)
...
#include « compute_auxiliary_bounds » ! Compute IstrU, IstrR
...
    
```

Fig. 5: Computation of auxiliary indexes

ATMOSPHERIC SURFACE BOUNDARY LAYER

Related CPP options:

BULK_FLUX	Activate bulk formulation for surface heat fluxes
BULK_FAIRALL	Choose Fairall formulation (default: COAMPS formulation)
BULK_EP	Add in bulk formulation for fresh water fluxes
BULK_LW	Add in long-wave radiation feedback from model SST
BULK_SMFLUX	Add in bulk formulation for surface momentum fluxes
SST_SKIN	Activate skin sst computation (Zeng & Beljaars, 2005)
ONLINE	Read native files and perform online interpolation on CROCO grid (default cubic interpolation)
QCORRECTION	Activate linearized bulk formulation providing heat flux correction dQdSST for nudging towards model SST
SFLX_CORR	Activate freshwater flux correction around model SSS
ANA_DIURNAL_SW	Activate analytical diurnal modulation of short wave radiations (only appropriate if there is no diurnal cycle in data)

Online CPP options:

ECMWF	Use ECMWF atm fluxes
AROME	Use METEO FRANCE fluxes
READ_PATM	Read atmospheric pressure instead of use default reference pressure and take into account atmospheric pressure gradient in the equations
OBC_PATM	In case of READ_PATM, inverse barometer effect to the open boundaries if atmospheric pressure is read in meteo file.

Preselected options:

ONLINE option is an alternative to pre-processing of surface forcing data, that can be useful for long-term simulations, especially if handling multiple configurations. ONLINE option calls for CUBIC_INTERP in set_global_definitions.h.

```
# BULK_FLUX
# ifdef BULK_FLUX
# define BULK_LW
# define BULK_EP
# define BULK_SMFLUX
# undef SST_SKIN
# undef ANA_DIURNAL_SW
# undef ONLINE
# ifdef ONLINE
#   define AROME
#   undef ERA_ECMWF
# endif
# define READ_PATM
# ifdef READ_PATM
```

(continues on next page)

(continued from previous page)

```
#  define OBC_PATM
#  endif
#  else
#  define QCORRECTION
#  define SFLX_CORR
#  define ANA_DIURNAL_SW
#  endif
```


OPEN BOUNDARIES CONDITIONS

11.1 OBC

Related CPP options:

OBC_EAST	Open eastern boundary
OBC_WEST	Open western boundary
OBC_SOUTH	Open southern boundary
OBC_NORTH	Open northern boundary

Related CPP options:

OBC_VOLCONS	Activate mass conservation enforcement at open boundaries (with OBC_M2ORLANSKI)
OBC_M2SPECIFIED	Activate specified open boundary conditions for barotropic velocities
OBC_M2ORLANSKI	Activate radiative open boundary conditions for barotropic velocities
OBC_M2FLATHER	Activate Flather open boundary conditions for barotropic velocities
OBC_M2CHARACT	Activate open boundary conditions based on characteristic methods barotropic velocities
OBC_M3SPECIFIED	Activate specified open boundary conditions for baroclinic velocities
OBC_M3ORLANSKI	Activate radiative open boundary conditions for baroclinic velocities
OBC_TSPECIFIED	Activate specified open boundary conditions for tracers
OBC_TORLANSKI	Activate radiative open boundary conditions for tracers
OBC_TUPWIND	Activate upwind open boundary conditions for tracers

For non-tidal forcing, the combination of OBC_M2ORLANSKI and OBC_VOLCONS often provides the best performances in terms of transparency of barotropic flow at the open boundaries. OBC_M2CHARACT is near as good and also provides proper conditions for tidal forcing.

Preselected options:

```
# ifdef TIDES
#   define OBC_M2FLATHER
# else
#   undef OBC_M2SPECIFIED
#   undef OBC_M2FLATHER
#   define OBC_M2CHARACT
#   undef OBC_M2ORLANSKI
#   ifdef OBC_M2ORLANSKI
#     define OBC_VOLCONS
#   endif
# endif
# define OBC_M3ORLANSKI
# define OBC_TORLANSKI
# undef OBC_M3SPECIFIED
# undef OBC_TSPECIFIED
```

11.2 Sponge Layer

SPONGE is preselected in cppdefs.h and calls for SPONGE_GRID in set_global_definitions.h. SPONGE_GRID selects the sponge layer extension (10 points with cosine shape function) and viscosity and diffusivity values according to the horizontal resolution.

Related CPP options:

SPONGE	Activate areas of enhanced viscosity and diffusivity near lateral open boundaries.
SPONGE_GRID	Automatic setting of the sponge width and value
SPONGE_DIF2	Sponge on tracers (default)
SPONGE_VIS2	Sponge on momentum (default)
SPONGE_SED	Sponge on sediment (default)

11.3 Nudging layers

Related CPP options:

ZNUDGING	Activate nudging layer for sea level
M2NUDGING	Activate nudging layer for barotropic velocities
M3NUDGING	Activate nudging layer for baroclinic velocities
TNUDGING	Activate nudging layer for tracers
ROBUST_DIAG	Activate nudging over the whole domain

The nudging layer has the same extension as the sponge layer. In nudging layers, tracer and momentum fields are nudged towards climatological values at a time scale τ_{out} (possibly different for momentum and tracers) that is given in namelist croco.in

11.4 Lateral forcing

11.4.1 CLIMATOLOGY strategy

Related CPP options:

CLIMATOLOGY	Activate processing of 2D/3D data (climatological or simulation/reanalysis) used as forcing at the open boundary points + nudging layers
ZCLIMATOLOGY	Activate processing of sea level
M2CLIMATOLOGY	Activate processing of barotropic velocities
M3CLIMATOLOGY	Activate processing of baroclinic velocities
TCLIMATOLOGY	Activate processing of tracers

11.4.2 BRY strategy

FRC_BRY is useful for inter-annual forcing on high-resolution domains. FRC_BRY is compatible with CLIMATOLOGY that can still be used for nudging layers.

Related CPP options:

FRC_BRY	Activate processing of 1D/2D data used as forcing at open boundary points strictly
Z_FRC_BRY	Activate open boundary forcing for sea level
M2_FRC_BRY	Activate open boundary forcing for barotropic velocities
M3_FRC_BRY	Activate open boundary forcing for baroclinic velocities
T_FRC_BRY	Activate open boundary forcing for tracers

Preselected options:

```
# define CLIMATOLOGY
# ifdef CLIMATOLOGY
# define ZCLIMATOLOGY
# define M2CLIMATOLOGY
# define M3CLIMATOLOGY
# define TCLIMATOLOGY
# define ZNUDGING
# define M2NUDGING
# define M3NUDGING
# define TNUDGING
# undef ROBUST_DIAG
# endif
# undef FRC_BRY
# ifdef FRC_BRY
# define Z_FRC_BRY
# define M2_FRC_BRY
# define M3_FRC_BRY
# define T_FRC_BRY
# endif
```


Related CPP options:

PSOURCE	Activate point sources (rivers)
ANA_PSOURCE	use analytical vertical profiles for point sources (set in set_global_definitions.h)
PSOURCE_NCFILE	Read variable river transports in netcdf file
PSOURCE_NCFILE_TS	Read variable river concentration in netcdf file

ANA_PSOURCE gives the vertical distribution of point source outflow. The default shape is an uniform distribution along the vertical. The vertical shape can be customized in subroutine *ana_psource* in *analytical.F*

An example of runoff file is given below

```
netcdf croco_runoff {
dimensions:
    qbar_time = 28193 ;
    n_qbar = 9 ;
    runoffname_StrLen = 30 ;
    two = 2 ;
    temp_src_time = 10690 ;
    salt_src_time = 10690 ;
variables:
    double qbar_time(qbar_time) ;
        qbar_time:long_name = "runoff time" ;
        qbar_time:units = "days" ;
        qbar_time:cycle_length = 0. ;
        qbar_time:long_units = "days since 1900-01-01" ;
    char runoff_name(n_qbar, runoffname_StrLen) ;
        runoff_name:long_name = "runoff name" ;
    double runoff_position(n_qbar, two) ;
        runoff_position:long_name = "position of the runoff (by line) in the
↔CROCO grid" ;
    double runoff_direction(n_qbar, two) ;
        runoff_direction:long_name = "direction/sense of the runoff (by
↔line) in the CROCO grid" ;
    double Qbar(n_qbar, qbar_time) ;
        Qbar:long_name = "runoff discharge" ;
        Qbar:units = "m3.s-1" ;
    double temp_src_time(temp_src_time) ;
        temp_src_time:cycle_length = 0. ;
        temp_src_time:long_units = "days since 1900-01-01" ;
    double salt_src_time(salt_src_time) ;
        salt_src_time:cycle_length = 0. ;
        salt_src_time:long_units = "days since 1900-01-01" ;
```

(continues on next page)

```

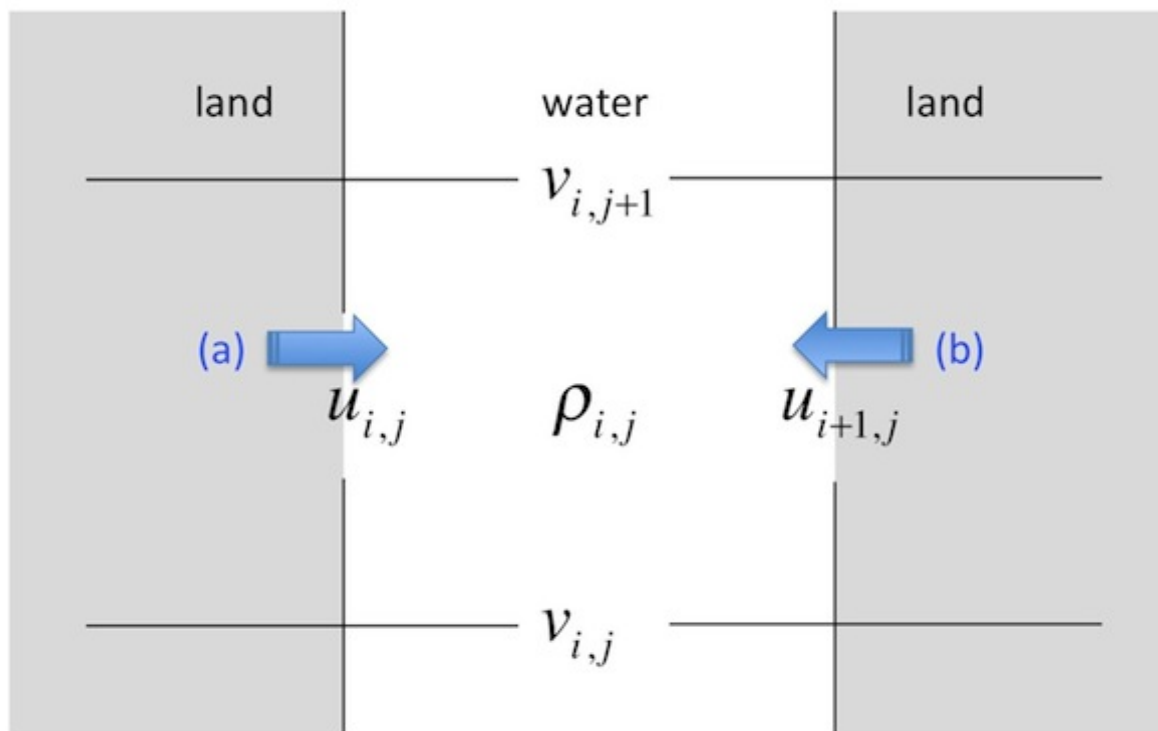
double temp_src(n_qbar, temp_src_time) ;
    temp_src:long_name = "runoff temperature" ;
    temp_src:units = "Degrees Celcius" ;
double salt_src(n_qbar, temp_src_time) ;
    salt_src:long_name = "runoff salinity" ;
    salt_src:units = "psu" ;
}
    
```

When using PSOURCE, Isrc and Jsrc refer to the i,j index of the u-face or v-face the flow crosses - NOT the i,j index of the rho cell it flows into. The i,j values must follow ROMS Fortran numbering convention for the appropriate u-point or v-point on the ROMS staggered grid.

This numbering convention is shown in the figure below (Courtesy of ROMS-RUTGERS team) for flow crossing a u-face into a cell from either the left or the right. This makes it more obvious why the index of the u-face must be specified, because to give the i,j indices of the receiving rho-cell would be ambiguous.

The u-face or v-face should be a land/sea mask boundary (i.e. a coastline). If the cell face is placed wholly in the land you get nothing because there is no wet cell for the flow to enter. If the face is in the middle of open water you have a situation where the flow at that cell face computed by the advection algorithm is 'REPLACED, not augmented, by the source.

It is very easy to misconfigure source/sink locations so caution and careful checking is required.



TIDES

Related CPP options:

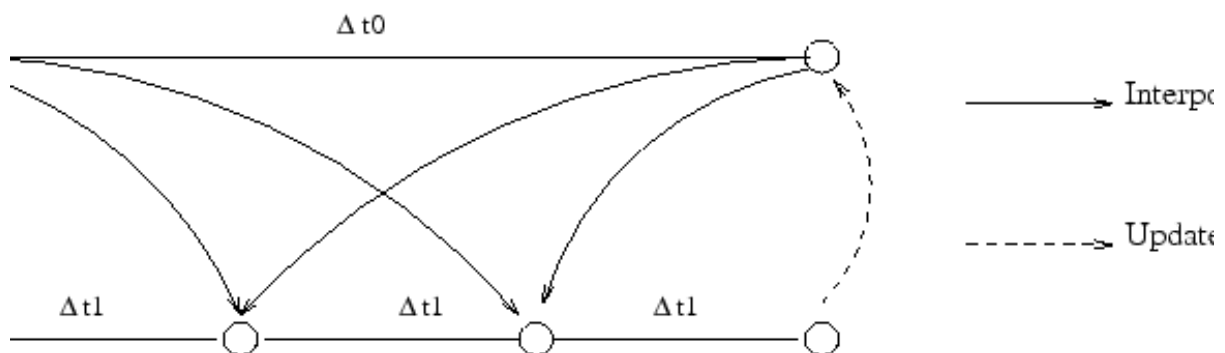
TIDES	Activate tidal forcing at open boundaries
SSH_TIDES	process and use tidal sea level data
UV_TIDES	process and use tidal current data
OBC_REDUCED_PHYSICS	compute tidal velocity from tidal elevation in case of tidal current is not available
TIDERAMP	Apply ramping on tidal forcing (1 day) at initialization Warning! This should be undefined if restarting the model

Preselected options:

```
# ifdef TIDES
#   define SSH_TIDES
#   define UV_TIDES
#   ifndef UV_TIDES
#     define OBC_REDUCED_PHYSICS
#   endif
#   define TIDERAMP
# endif
```


NESTING CAPABILITIES

To address the challenge of bridging the gap between near-shore and offshore dynamics, a nesting capability has been added to CROCO and tested for the California Upwelling System (Debreu et al., 2012; Penven et al., 2006). The method chosen for embedded gridding takes advantage of the AGRIF (Adaptive Grid Refinement in Fortran) package (Debreu and Blayo, 2003, 2008; Blayo and Debreu, 1999; Debreu and Vouland, 2003; Debreu, 2000). AGRIF is a Fortran 95 package for the inclusion of adaptive mesh refinement features within a finite difference numerical model. One of the major advantages of AGRIF in static-grid embedding is the ability to manage an arbitrary number of fixed grids and an arbitrary number of embedding levels.



A recursive integration procedure manages the time evolution for the child grids during the time step of the parent grids (Fig. 2). In order to preserve the CFL criterion, for a typical coefficient of refinement (say, a factor of 3 for a 5 km resolution grid embedded in a 15 km grid), for each parent time step the child must be advanced using a time step divided by the coefficient of refinement as many time as necessary to reach the time of the parent (Fig. 2). For simple 2-level embedding, the procedure is as follows:

1. Advance the parent grid by one parent time step.
2. Interpolate the relevant parent variables in space and time to get the boundary conditions for the child grid.
3. Advance the child grid by as much child time steps as necessary to reach the new parent model time.
4. Update point by point the parent model by averaging the more accurate values of the child model (in case of 2-way nesting).

The recursive approach used in AGRIF allows the specification of any number of nesting levels. Additional CPP options are related to AGRIF, they are in set_global_definitions.h and set_obc_definitions.h files. These are default options intended for nesting developers and should not be edit by standard users.

For a better understanding of ROMS nesting capabilities using AGRIF, check the published articles on CROCO/ROMS nesting implementation and also the AGRIF project homepage:

1. CROCO/ROMS 1 way nesting : Evaluation and application of the ROMS 1-way, embedding procedure to the central california upwelling system, (Penven et al., 2006)
2. CROCO/ROMS 2 way nesting: Two-way nesting in split-explicit ocean models: algorithms, implementation and validation. (Debreu et al., 2012)
3. AGRIF homepage : <http://www-ljk.imag.fr/MOISE/AGRIF/>

Related CPP options:

AGRIF	Activate nesting capabilities (1-WAY by default)
AGRIF_2WAY	Activate 2-WAY nesting (update parent solution by child solution)

SEDIMENT AND BIOLOGY MODELS

15.1 Bottom Boundary Layer model

Related CPP options:

BBL	Activate bottom boundary layer parametrization
ANA_WWAVE	Set analytical (constant) wave forcing (hs, Tp, Dir).
ANA_BSEDIM	Set analytical bed parameters (if SEDIMENT is undefined)
Z0_BL	Compute bedload roughness for ripple predictor and sediment purposes
Z0_RIP	Determine bedform roughness ripple height and ripple length for sandy bed
Z0_BIO	Determine (biogenic) bedform roughness ripple height and ripple length for silty beds

Preselected options:

```
#ifndef BBL
# ifdef OW_COUPLING
# elif defined WAVE_OFFLINE
# elif defined WKB_WWAVE
# else
# define ANA_WWAVE
# endif
# ifdef SEDIMENT
# undef ANA_BSEDIM
# else
# define ANA_BSEDIM
# endif
# ifdef SEDIMENT
# define Z0_BL
# else
# undef Z0_BL
# endif
# ifdef Z0_BL
# define Z0_RIP
# endif
# undef Z0_BIO
#endif
```

DESCRIPTION

Reynolds stresses, production and dissipation of turbulent kinetic energy, and gradients in velocity and suspended-sediment concentrations vary over short vertical distances, especially near the bed, and can be difficult to resolve with the vertical grid spacing used in regional-scale applications. CROCO provides algorithms to parameterize some of these subgrid-scale processes in the water column and in the bottom boundary layer (BBL). Treatment of the BBL is important for the circulation model solution because it determines the stress exerted on the flow by the bottom, which enters the Reynolds-averaged Navier-Stokes equations as a boundary conditions for momentum in

the x and y directions:

$$K_m \frac{\partial u}{\partial s} = \tau_{bx}$$

$$K_m \frac{\partial v}{\partial s} = \tau_{by}$$

Determination of the BBL is even more important for the sediment-transport formulations because bottom stress determines the transport rate for bedload and the resuspension rate for suspended sediment.

CROCO implements either of two methods for representing BBL processes: (1) simple drag-coefficient expressions or (2) more complex formulations that represent the interactions of wave and currents over a moveable bed. The drag-coefficient methods implement formulae for linear bottom friction, quadratic bottom friction, or a logarithmic profile. The other, more complex wave-current BBL model is described by Blaas et al. (2007) with an example of its use on the Southern California continental shelf. The method uses efficient wave-current BBL computations developed by Soulsby (1995) in combination with sediment and bedform roughness estimates of Grant and Madsen (1982), Nielsen (1986) and Li and Amos (2001).

Linear/quadratic drag

The linear and/or quadratic drag-coefficient methods depend only on velocity components u and v in the bottom grid cell and constant, spatially-uniform coefficients γ_1 and γ_2 specified as input:

$$\tau_{bx} = (\gamma_1 + \gamma_2 \sqrt{u^2 + v^2}) u$$

$$\tau_{by} = (\gamma_1 + \gamma_2 \sqrt{u^2 + v^2}) v$$

where γ_1 is the linear drag coefficient and γ_2 is the quadratic drag coefficient. The user can choose between linear or quadratic drag by setting one of these coefficients to zero. The bottom stresses computed from these formulae depend on the elevation of u and v (computed at the vertical mid-elevation of the bottom computational cell). Therefore, in this s -coordinate model, the same drag coefficient will be imposed throughout the domain even though the vertical location of the velocity is different.

Logarithmic drag (with roughness length z_0)

To prevent this problem, the quadratic drag γ_2 can be computed assuming that flow in the BBL has the classic vertical logarithmic profile defined by a shear velocity u_* and bottom roughness length z_0 (m) as:

$$|u| = \frac{u_*}{\kappa} \ln \left(\frac{z}{z_0} \right)$$

where $|u| = \sqrt{u^2 + v^2}$, friction velocity $u_* = \sqrt{\tau_b}$, z is the elevation above the bottom (vertical mid-elevation point of the bottom cell), $\kappa = 0.41$ is von Kármán's constant. z_0 is an empirical parameter. It can be constant (default) or spatially varying. Kinematic stresses are calculated as`

$$\tau_{bx} = \frac{\kappa^2}{\ln^2(z/z_0)} \sqrt{u^2 + v^2} u$$

$$\tau_{by} = \frac{\kappa^2}{\ln^2(z/z_0)} \sqrt{u^2 + v^2} v$$

The advantage of this approach is that the velocity and the vertical elevation of that velocity are used in the equation. Because the vertical elevation of the velocity in the bottom computational cell will vary spatially and temporally, the inclusion of the elevation provides a more consistent formulation.

Combined wave-current drag (BBL)

To provide a more physically relevant value of z_0 , especially when considering waves and mobile sediments, a more complex formulation is available (BBL).

The short (order 10-s) oscillatory shear of wave-induced motions in a thin (a few cm) wave-boundary layer produces turbulence and generates large instantaneous shear stresses. The turbulence enhances momentum transfer, effectively increasing the bottom-flow coupling and the frictional drag exerted on the wave-averaged flow. The large instantaneous shear stresses often dominate sediment resuspension and enhance bedload transport. Sediment transport can remold the bed into ripples and other bedforms, which present roughness elements to the flow. Bedload transport can also induce drag on the flow, because momentum is transferred to particles as they are removed

from the bed and accelerated by the flow. Resuspended sediments can cause sediment-induced stratification and, at high concentrations, change the effective viscosity of the fluid.

The BBL parameterization implemented in CROCO requires inputs of velocities u and v at reference elevation z , representative wave-orbital velocity amplitude u_b , wave period T , and wave propagation direction θ (degrees, clockwise from north). The wave parameters may be the output of a wave model such as WKB or WW3 or simpler calculations based on specified surface wave parameters. Additionally the BBL models require bottom sediment characteristics (median grain diameter D_{50} , mean sediment density ρ_s , and representative settling velocity w_s); these are constant (ANA_BSEDIM) or based on the composition of the uppermost active layer of the bed sediment during the previous time step if the sediment model is used.

The wave-averaged, combined wave-current bottom stress is expressed as function of τ_w and τ_c (i.e., the stress due to waves in the absence of currents and due to currents in the absence of waves, respectively) according to Soulsby (1995):

$$\bar{\tau}_{wc} = \tau_c \left(1 + 1.2 \left(\frac{\tau_w}{\tau_w + \tau_c} \right)^{3.2} \right)$$

The maximum wave-current shear stress within a wave cycle is obtained by adding $\bar{\tau}_{wc}$ and τ_w (with ϕ the angle between current and waves):

$$\tau_{wc} = \left((\bar{\tau}_{wc} + \tau_w \cos \phi)^2 + (\tau_w \sin \phi)^2 \right)^{1/2}$$

The stresses τ_c and τ_w are determined using:

$$\begin{aligned} \tau_c &= \frac{\kappa^2}{\ln^2(z/z_0)} |u|^2 \\ \tau_w &= 0.5 \rho f_w u_b^2 \end{aligned}$$

u_b , the bottom orbital velocity, is determined from the significant wave height H_s and peak frequency ω_p using the Airy wave theory:

$$u_b = \omega_p \frac{H_s}{2 \sinh kh}$$

with h the local depth and k the local wave number from the dispersion relation. The wave-friction factor f_w is, according to Soulsby (1995):

$$f_w = 1.39 (u_b / \omega_p z_0)^{-0.52}$$

The wave-current interaction in the BBL is taken into account only if $u_b > 1$ cm/s; otherwise, current-only conditions apply.

Shear stress for sediment resuspension and roughness length due to bed form

To determine the shear stress relevant for sediment resuspension and the roughness length due to bed forms, we follow the concept of Li and Amos (2001) briefly summarized here. First, the maximum wave-current skin friction τ_s is computed from the equations above, using the Nikuradse roughness $z_0 = D_{50}/12$.

A bed-load layer develops as soon as the maximum wave-current skin friction τ_s exceeds the critical stress τ_{cr} . This layer affects the stress effective for ripple formation and sediment resuspension. Subsequently, for sandy locations, ripple height and length are computed, leading to a space- and time-dependent ripple roughness length $z_0 = z_{rip}$, which is used to compute the drag on the flow (instead of a constant value when BBL is not activated). This drag provides boundary conditions to the momentum and turbulence equations (KPP or GLS).

15.2 Sediment models

There are two sediment models in CROCO: the USGS model derived from the UCLA/USGS ROMS community, and MUSTANG derived from the Ifremer SIAM/MARS community.

15.2.1 USGS Sediment Model

This USGS sediment model is derived from the UCLA/USGS ROMS community. See Blaas et al. (2007) and Warner et al. (2008) or the report given in the documentation section for details.

Regarding the time and space resolution considered, the explicit solution generally refers to quantities averaged over wave periods, although the implementation of a nonhydrostatic solver in CROCO opens the way to a wave-resolved approach. One of the crucial ingredients in the sediment transport model is a reliable representation of wave-averaged (or wave-resolved) hydrodynamics and turbulence.

In the wave-averaged approach, the wave boundary layer is not resolved explicitly, but the lower part of the velocity and sediment concentration profile in the current boundary layer is important for the calculation of the sediment transport rates. Similarly, an accurate assessment of the bottom boundary shear stress (including effect of waves) is required since it determines the initiation of grain motion and settling and resuspension of suspended load (see BBL). Thus, the sediment concentration and current velocity profiles in the unresolved part of the near-bottom layer have to be parameterized. Characterization of the sediments (mainly density and grain size, making general assumptions about shape and cohesiveness) is done either as a time-dependent prescribed function at the point sources or at the sea bed as an initial (soon space-dependent) condition. Sediment concentration may be considered as passive with respect to the flow density or as active if concentration values require such (the latter is not implemented yet).

Sediment bed

The sediment bed is represented by three-dimensional arrays with a fixed number of layers beneath each horizontal model cell. Each cell of each layer in the bed is initialized with a thickness, sediment-class distribution, porosity, and age. The mass of each sediment class in each cell can be determined from these values and the grain density. The bed framework also includes two-dimensional arrays that describe the evolving properties of the seabed, including bulk properties of the surface layer (active layer thickness, mean grain diameter, mean density, mean settling velocity, mean critical stress for erosion) and descriptions of the subgrid scale morphology (ripple height and wavelength). These properties are used to estimate bed roughness in the BBL formulations and feed into the bottom stress calculations. The bottom stresses are then used by the sediment routines to determine resuspension and transport, providing a feedback from the sediment dynamics to the hydrodynamics.

The bed layers are modified at each time step to account for erosion and deposition and track stratigraphy. At the beginning of each time step, an active layer thickness z_a is calculated (Harris and Wiberg, 1997). z_a is the minimum thickness of the top bed layer. If the top layer is thicker than z_a , no action is required. If the top layer is less than z_a , then the top layer thickness is increased by entraining sediment mass from deeper layers until the top layer thickness equals z_a . If sediment from deeper than the second layer is mixed into the top layer, the bottom layer is split to enforce a constant number of layers and conservation of sediment mass. Each sediment class can be transported by suspended-load and/or bedload (below). Suspended-load mass is exchanged vertically between the water column and the top bed layer. Mass of each sediment class available for transport is limited to the mass available in the active layer. Bedload mass is exchanged horizontally within the top layer of the bed. Mass of each sediment class available for transport is limited to the mass available in the top layer. Suspended-sediment that is deposited, or bedload that is transported into a computational cell, is added to the top bed layer. If continuous deposition results in a top layer thicker than a user-defined threshold, a new layer is provided to begin accumulation of depositing mass. The bottom two layers are then combined to conserve the number of layers. After erosion and deposition have been calculated, the active-layer thickness is recalculated and bed layers readjusted to accommodate it. This step mixes away any very thin layer (less than the active layer thickness) of newly deposited material. Finally the surficial sediment characteristics, such as D_{50} , ripple geometry, etc., are updated and made available to the bottom stress calculations.

Suspended-sediment transport

Sediment suspended in the water column is transported, like other conservative tracers (e.g., temperature and salinity) by solving the advection–diffusion equation with a source/sink term for vertical settling and erosion. For each sediment size class j :

$$\frac{\partial c_j}{\partial t} + \frac{\partial u_i c_j}{\partial x_i} - \frac{\partial}{\partial x_i} \left(K_i \frac{\partial c_j}{\partial x_i} \right) - w_{sj} \frac{\partial c_j}{\partial x_3} = Q_j$$

Subscript i denotes coordinate direction (with x_3 vertically upward) and repeated indices imply summation; c_j is the Reynolds-averaged, wave-averaged sediment concentration of sediments in class j ; u_i is velocity; K_i is the eddy diffusivity; w_{sj} is the settling velocity, dependent on sediment grain size, but independent of flow conditions and concentrations; and Q_j represents point sources or sinks. Zero-flux boundary conditions are imposed at the surface and bottom in the vertical diffusion equation. Lateral open boundaries are treated as other tracers according to Marchesiello et al. (2001). A quasi-monotonic 5th-order advection scheme (WENO5-Z, Borges et al., 2008) can be used for horizontal and vertical advection of all tracers, including sediments.

The settling velocity w_s is an input parameter of the model (WSED in sediment.in; see below). Settling is computed via a semi-Lagrangian advective flux algorithm. It uses a piece-wise parabolic, vertical reconstruction of the suspended sediment in the water column with PPT/WENO constraints to avoid oscillations. This method allows the integration bounds of depositional flux to use multiple grid boxes in the vertical direction, so it is not constrained by the CFL criterion.

For each size class, the source or sink term represents the net of upward flux of eroded material E and downward settling, i.e., the deposition flux ($-w_s c$). The erosion flux for each class is given by:

$$\begin{aligned} E &= E_0(1-p)\phi \left(\frac{\tau_s}{\tau_{cr}} - 1 \right) & \text{for } \tau_s > \tau_{cr} \\ E &= 0 & \text{for } \tau_s < \tau_{cr} \end{aligned} \quad (15.1)$$

E_0 is an empirical erosion rate (ERATE parameter in sediment.in; see below); p is the sediment porosity; ϕ is the volumetric fraction of sediment of the class considered; τ_{cr} is the critical shear stress; and τ_s is the shear stress magnitude on the grains (skin stress due to wave-induced bed orbital velocities and mean bottom currents; see BBL). The critical shear stress is the threshold for the initiation of sediment motion.

Bedload transport

This version implements two methods for computing bedload transport: 1) the Meyer-Peter Müller (1948) formulation for unidirectional flow and 2) the formulae of Soulsby and Damgaard (2005) that accounts for combined effects of currents and symmetric and asymmetric waves. The formulae depend on the characteristics of individual sediment classes, including median size D_{50} , grain density ρ_s , specific density in water $s = \rho/\rho_s$, and critical shear stress τ_c . Non-dimensional transport rates Φ are calculated for each sediment class and converted to dimensional bedload transport rates Q_{bl} using:

$$Q_{bl} = \Phi \sqrt{(s-1)gD_{50}^3\rho_s}$$

These are horizontal vector quantities with directions that correspond to the combined bed-stress vectors. Details on the computation of Φ differs in the Meyer-Peter Müller or Soulsby and Damgaard formulations.

Meyer-Peter Müller (1948)

$$\Phi = \max [8(\theta_s - \theta_c)^{1.5}, 0]$$

where Φ is the magnitude of the non-dimensional transport rate for each sediment class, θ_s is the non-dimensional Shields parameter for skin stress:

$$\theta_s = \frac{\tau_s}{(s-1)gD_{50}}$$

θ_c is the critical Shields parameter, and τ_s the magnitude of total skin-friction component of bottom stress computed from:

$$\tau_s = \sqrt{\tau_{sx}^2 + \tau_{sy}^2}$$

where τ_{sx} and τ_{sy} are the skin-friction components of bed stress, from currents alone or the maximum wave-current combined stress, in the x and y directions. These are computed at cell faces (u and v locations) and then interpolated to cell centers (ρ points). The bedload transport vectors are partitioned into x and y components based on the magnitude of the bed shear stress as:

$$Q_{blx} = Q_{bl} \frac{\tau_{sx}}{\tau_s}$$

$$Q_{bly} = Q_{bl} \frac{\tau_{sy}}{\tau_s}$$

Soulsby and Damgaard (2005)

$$\vec{\Phi} = \max \left[A_2 \theta_s^{0.5} (\theta_s - \theta_c) \frac{\vec{\theta}_s}{\theta_s}, 0 \right]$$

where $\vec{\Phi}$ and $\vec{\theta}_s$ are vectors with components in the direction of the mean current and in the direction perpendicular to the current. $A_2 = 12$ is a semi-empirical coefficient. The implementation of this method requires computation of transport rates in the directions parallel and perpendicular to the currents as:

$$\Phi_{\parallel} = \max [\Phi_{\parallel 1}, \Phi_{\parallel 2}]$$

where

$$\Phi_{\parallel 1} = A_2 \theta_m^{0.5} (\theta_m - \theta_c)$$

$$\Phi_{\parallel 2} = (0.9534 + 0.1907 \cos 2\phi) \theta_w^{0.5} \theta_m + A_2 (0.229 \gamma_w \theta_w^{0.5} \cos \phi)$$

$$\Phi_{\perp} = A_2 \frac{0.1907 \theta_w^2}{\theta_w^{1.5} + 1.5 \theta_m^{1.5}} (\theta_m \sin 2\phi + 1.2 \gamma_w \theta_w \sin \phi)$$

where ϕ is the angle between waves and currents. θ_m is the mean Shields parameter from τ_m as:

$$\tau_m = \tau_c \left(1 + 1.2 \left(\frac{\tau_w}{\tau_w + \tau_c} \right)^{1.5} \right)$$

τ_c is the bottom stress from the currents only, τ_w is the bottom stress from the waves only calculated in the bottom-boundary layer routines (see below).

Note: the asymmetry factor γ_w is the ratio between the amplitude of the second harmonic and the amplitude of the first harmonic of the oscillatory wave stress. Following the suggestion of Soulsby and Damgaard (2005), we estimate the asymmetry factor using Stokes second-order theory (e.g., Fredsøe and Deigaard, 1992) and constrain it to be less than 0.2. The non-dimensional fluxes are rotated into x and y directions using the directions for mean current and waves and dimensionalized to yield values for Q_{blx} and Q_{bly} for each sediment class.

Morphology

The bed model accounts for changes in sea floor elevation resulting from convergence or divergence in sediment fluxes (Exner equation). These morphological changes can have significant influence on flow and transport when they are larger than a few percent of the water depth. The morphological changes are accounted for by equating the bottom boundary condition of the vertical velocity to the rate of change of elevation of the sea floor. This method is completely mass conserving and retains tracer constancy preservation.

A morphological scale factor is also provided to allow an increased rate of morphological change, which can be useful for simulating evolution over long time periods. Strategies for morphological updating are described by

Roelvink (2006). In our implementation, bedload fluxes, erosion, and deposition rates are multiplied by a scale factor. A scale factor with a value of one has no effect, and values greater than one accelerate the bed response. For bedload transport, the scale factor is multiplied against the bedload transport rates. For suspended load transport, the scale factor multiplies the exchange of sediment (erosive or depositional flux) at the bed-water interface. The magnitude of sediment concentrations in the water column are not modified – just the exchange rate to and from the bed. For both bedload and suspended load, sediment is limited in availability as described previously, based on the true amount of sediment mass (not multiplied by the scale factor). This morphological scale factor method works well for systems with unlimited sediment in the bed. However, it can generate extra sediment in systems with limited supplies of bed sediment. This occurs when the amount of sediment to be eroded is limited by the amount available and application of the morphological scale factor cannot remove the scaled amount of sediment from the bed. Subsequent deposition does place a scaled amount of sediment on the bed thus creating new mass in the bed. Other approach (Lesser et al 2004) is to limit the amount of sediment fluxed to the water column in these situations. This gives unrealistically low sediment concentrations, but conserves bed sediment.

Sediment Density

This is not implemented yet. Effects of suspended sediment on the density field can be included with terms for the weight of each sediment class in the equation of state for seawater density as:

$$\rho = \rho_{water} + \sum_{m=1}^{N_{sed}} \frac{C_m}{\rho_{s,m}} (\rho_{s,m} - \rho_{water})$$

This enables the model to simulate processes where sediment density influences hydrodynamics, such as density stratification and gravitationally driven flows.

Related CPP options:

ANA_SEDIMENT	Set analytical sediment ripple and bed parameters
ANA_WWAVE	Analytical (constant) wave (hs, Tp, Dir) values.
SUSPLOAD	Activate suspended load transport
BEDLOAD	Activate bedload transport
MORPHODYN	Activate morphodynamics
ANA_BPFLUX	Set kinematic bottom flux of sediment tracer (if different from 0)
SLOPE_NEMETH	Nemeth formulation for avalanching (Nemeth et al, 2006)
SLOPE_LESSER	Lesser formulation for avalanching (Lesser et al, 2004)
BED-LOAD_SOULSBY	Soulsby formulation for bedload (Soulsby, R.L. and J.S. Damgaard, 2005)
BEDLOAD_MPM	Meyer-Peter-Muller formulation for bedload (Meyer-Peter, E. and R. Muller, 1948)

Preselected options:

```
#ifndef SEDIMENT
# define SUSPLOAD
# define BEDLOAD
# ifdef BEDLOAD
# undef SLOPE_NEMETH
# define SLOPE_LESSER
# if (defined WAVE_OFFLINE || defined WKB_WWAVE || defined ANA_WWAVE \
|| defined OW_COUPLING)
# define BEDLOAD_SOULSBY
# define Z0_BL
# define Z0_RIP
# else
# define BEDLOAD_MPM
# endif
# endif
# undef MORPHODYN
#endif
```

Parameters in sediment.in

```

1  Stitle (a80)
CROCO - Sediment - Test

2  Sd(1-NST), CSED, SRHO, WSED, ERATE, TAU_CE, TAU_CD, BED_FRAC(1:NLAY)
    0.125  9.9  2650.  9.4  25.0e-5  0.05  0.14  0.4  0.4
    0.050  0.0  2650.  1.6  4.0e-5  0.01  0.14  0.6  0.6

3  BTHK(1:NLAY)
    1.  10.

4  BPOR(1:NLAY)
    0.41  0.42

5  Hrip
    0.03

6  Lrip
    0.14

7  bedload_coeff
    0.

8  morph_fac
    10.

99 END of sediment input data
    
```

GLOSSARY

** CARD 1: String with a maximum of eighty characters. **

Stitle Sediment case title.

** CARD 2: Sediment grain parameters & initial values (NST lines) **

Sd Diameter of grain size class [mm]. —

CSED Initial concentration (spatially uniform) [kg/m3]. —

SRHO Density of sediment material of size class [kg/m3]. Quartz: SRHO=2650 kg/m³

WSED Settling velocity of size class [mm/s]. Typically (Soulsby, 1997):

$$WSED = 10^3 (visc (\sqrt{10.36^2 + 1.049D^3} - 10.36) / D_{50} \text{ [mm/s]})$$

with $D = D_{50} (g (SRHO/\rho_0 - 1) / (visc^2))^{0.33333}$

$$D_{50} = 10^{-3} Sd \text{ [m]}$$

$$visc = 1.3 \cdot 10^{-3} / \rho_0 \text{ [m2/s]}$$

ERATE Erosion rate of size class [kg/m2/s]. Typically:

$$ERATE = 10^{-3} \gamma_0 WSED SRHO \text{ [kg/m}^2\text{/s]}$$

with $\gamma_0 = 10^{-3} - 10^{-5}$ (Smith & McLean, 1977)

TAU_CE Critical shear stress for sediment motion [N/m2] (initiation of bedload for coarses, suspension for fines). Typically :

$$TAU_CE = \theta_c g D_{50} (SRHO - \rho_0) \text{ [N/m2]} \text{ (bedload)}$$

with $\theta_c = 0.3 / (1 + 1.2 D) + 0.055 (1 - e^{-0.02 \cdot D})$ (Soulsby & Whitehouse 97)

$$TAU_CE = 6.4 \cdot 10^{-7} \rho_0 WSED^2 \text{ [N/m2]} \text{ (suspended load)}$$

TAU_CD Critical shear stress for deposition of cohesive sediments [N/m2] —

BED_FRAC Volume fraction of each size class in each bed layer (NLAY columns)
[0<BED_FRAC<1]

** CARD 3: Sediment bed thickness, 1st field is top layer ('delt_a') **

BTHK Initial thicknesses of bed layers [m] Bthk(1) active layer thickness, fixed in simulation unless SUM(Bthk(:))<Bthk(1)

** CARD 4: Sediment bed porosity **

BPOR Initial porosity of bed layers [m] used in ana_sediment ifdef ANA_SEDIMENT (not in init.nc)

** CARD 5: Bottom ripple height **

Hrip Initial ripple height [m] used in ana_sediment ifdef ANA_SEDIMENT (not in init.nc)

** CARD 6: Bottom ripple length **

Lrip Initial ripple length [m] used in ana_sediment ifdef ANA_SEDIMENT (not in init.nc)

15.2.2 MUSTANG Sediment model

15.3 Biogeochemical models

CROCO comes with series of biogeochemical (BGC) models of increasing complexity, from relatively simple 5- or 7-component NPZD (Gruber et al., 2006, 2011) and N2P2Z2D2 BioEBUS model (Gutknecht, 2013) that proved well suited to upwelling regions to 24-component PISCES (Aumont et al., 2005).

BioEBUS is a nitrogen-based model (Fig. 1) derived from a N2P2Z2D2 evolution of ROMS NPZD model (Gruber et al., 2006, 2011) and accounting for the main planktonic communities in upwelling ecosystems associated oxygen minimum zones (OMZs). It is validated in Gutknecht et al. (2013) using available satellite and in situ data in the northern part of the Benguela upwelling system. In this model, phytoplankton and zooplankton are split into small (PS and ZS: flagellates and ciliates, respectively) and large (PL and ZL: diatoms and copepods, respectively) organisms. Detritus are also separated into small and large particulate compartments (DS and DL). A semi-labile dissolved organic nitrogen (DON) compartment was added since DON can be an important reservoir of OM and can potentially play an important role in supplying nitrogen or carbon from the coastal region to the open ocean (Huret et al., 2005). The pool of dissolved inorganic nitrogen is split into nitrate (NO₃⁻), nitrite (NO₂⁻) and ammonium (NH₄⁺) species to have a detailed description of the microbial loop: ammonification/nitrification processes under oxic conditions, and denitrification/anammox processes under suboxic conditions (Yakushev et al., 2007). These processes are directly oxygen dependent, so an oxygen (O₂) equation was also introduced in BioEBUS with the source term (photosynthesis), sink terms (zooplankton respiration, bacteria re-mineralisation) and sea-air O₂ fluxes following Pena et al. (2010) and Yakushev et al. (2007). To complete this nitrogen-based model, nitrous oxide (N₂O) was introduced using the parameterization of Suntharalingam et al. (2000, 2012). It allows determining the N₂O production under oxygenated conditions and at low-oxygen levels, mimicking the N₂O production from nitrification and denitrification processes. The SMS terms of BioEBUS and parameter values are described in detail in Gutknecht et al. (2013).

PISCES was developed for NEMO (the French ocean climate model). It was implemented in CROCO for its supposed suitability for a wide range of oceanic regimes. PISCES currently has five modeled limiting nutrients for phytoplankton growth: Nitrate and Ammonium, Phosphate, Silicate and Iron. Phosphate and nitrate+ammonium are linked by constant Redfield ratios but the nitrogen pool undergoes nitrogen fixation and denitrification. Four living compartments are represented: two phytoplankton size-classes/groups corresponding to nanophytoplankton and diatoms, and two zooplankton size classes which are micro-zooplankton and mesozooplankton. For phytoplankton, prognostic variables are total biomass, the iron, chlorophyll and silicon contents. This means that the Fe/C, Chl/C and Si/C ratios of both phytoplankton groups are fully predicted by the model. For zooplankton, only the total biomass is modeled. For all species, the C/N/P/O₂ ratios are supposed constant and are not allowed to vary. The Redfield ratio O/C/N/P is set to 172/122/16/1. In addition, the Fe/C ratio of both zooplankton groups is kept constant. No silicified zooplankton is assumed. The bacterial pool is not yet explicitly modeled. There are three non-living compartments: semi-labile dissolved organic matter, small and big sinking particles. The iron, silicon and calcite pools of the particles are explicitly modeled and their ratios are allowed to vary. The sinking

speed of the particles is not altered by their content in calcite and biogenic silicate ("The ballast effect"). The latter particles are assumed to sink at the same speed as big organic matter particles. All the non-living compartments experience aggregation due to turbulence and differential settling. In addition to the ecosystem model, PISCES also simulates dissolved inorganic carbon, total alkalinity and dissolved oxygen. The latter tracer is also used to define the regions where oxic or anoxic remineralization takes place. see Aumont et al. (2005) in the documentation section for details.

Related CPP options:

PISCES	Activate 24-component PISCES biogeochemical model
BIO_NChlPZD	Activate 5-component NPZD type model
BIO_N2PZD2	Activate 7-component NPZD type model
BIO_BioEBUS	Activate 12-component NPZD type model

Preselected options:

```
# ifdef BIOLOGY
# undef PISCES
# define BIO_NChlPZD
# undef BIO_N2ChlPZD2
# undef BIO_BioEBUS
# endif
```

15.4 Lagrangian floats

COUPLING CROCO WITH OTHER MODELS

CROCO is coupled to atmospheric and wave models through the OASIS-MCT (Ocean-Atmosphere-Sea-Ice-Soil, Model Coupling Toolkit) coupler developed by CERFACS (Toulouse, France). This coupler allows the atmospheric, oceanic, and wave models to run at the same time in **parallel**, it **exchanges** variables, and performs **grid interpolations** and **time transformations** if requested. OASIS is not an executable file, but a set of libraries providing functions which are called in the models themselves. The variables exchanged by the coupler, as well as the grid interpolations are specified through a namelist file (called `namcouple`).

CROCO can therefore be coupled to any code in which OASIS-MCT is implemented. Non-exhaustively, here are some models including OASIS-MCT, that can be coupled to CROCO:

- WRF (Weather Research and Forecast model developed at NCAR, Boulder, USA)
- Meso-NH (Mesoscale Non-Hydrostatic model developed at Laboratoire d'Aérodynamique, Toulouse, France)
- WW3 (WaveWatch III model developed at NCEP, USA, and Ifremer, France)
- ...

Those model are not provided for download with CROCO and need to be installed separately, as well as OASIS-MCT library.

A description of the OASIS-MCT features, its implementation in CROCO, WW3 and WRF codes, and the coupled variables that can be exchanged are given in the following.

Detailed step by step coupled tutorial is also available in the **Tutorials** section.

16.1 OASIS philosophy

16.1.1 OASIS libraries

OASIS-MCT libraries are:

- **psmile** for coupling
- **mct** (Argonne National Laboratory) for parallel exchanges
- **scrip** (Los Alamos National Laboratory) for interpolations

Functions provided by the OASIS-MCT framework are:

Note: `oasis_`/`prism_` are new / old names for backward compatibility, both useable

- Initialization and creation of a local communicator for internal parallel computation in each model:
 - `oasis_init_comp` / `prism_init_comp_proto`
 - `oasis_get_localcomm` / `prism_get_localcomm_proto`
- Grid data definition for exchanges and interpolations:

- oasis_write_grid
- oasis_write_corner
- oasis_write_area
- oasis_write_mask
- oasis_terminate_grids_writing
- Partition and exchanged variables definition:
 - oasis_def_partition / prism_def_partition_proto
 - oasis_def_var / prism_def_var_proto
 - oasis_enddef / prism_enddef_proto
- Exchange of coupling fields:
 - oasis_get / prism_get_proto
 - oasis_put / prism_put_proto
- Finalization:
 - oasis_terminate / prism_terminate_proto

These OASIS3-MCT intrinsic functions are called in each model involved in the coupling. **Initialization** phase, **Definition** phase, and **Finalization** phase are called only once in each simulation while **Exchange** phase is called every time step. The effective exchanges are done only at specified times, defined by the coupling frequency, although the **Exchange** phase is called every model time step. The coupling frequency is controlled through the OASIS3-MCT namcouple.

16.1.2 Coupling sequence

The frequency of exchanges between two models is defined by the **coupling time step**.

The **coupling time step** must be a multiple of the models time steps. An example of coupling sequence is pictured in the following Figure. In this example, the coupling time step is defined at 360s for both models. The wave model time step is 90s, so it will exchange every 4 time steps. The ocean model time step is 180s, so it will exchange every 2 time steps.

Another coupling parameter defined in the namcouple is the **lag**. It is used by the OASIS coupler to synchronize the `send` and `receive` functions. The lag must be defined for each model at the same value than its own time step. For instance:

- WAVE to OCEAN lag = dt wave = 90
- OCEAN to WAVE lag = dt ocean = 180

16.1.3 Restart files

As reception of coupled fields is called before model computation, you need to create **restart files** for the coupler containing initial or restart fields for the first time step.

These **restart files** are for OASIS, and therefore need to have variable names corresponding to OASIS `namcouple` coupled fields. The initial files for OASIS are named `oasis_oce.nc` and `oasis_wave.nc` in the example pictured in the above Figure. `oce_ini` and `wave_ini` are not related to OASIS, they are usual initialization or restart files from your oceanic and wave model; e.g. in CROCO, `oce_ini` is `croco_ini.nc`, and in WW3, `wave_ini` is `restart.wv3`.

Summary of the restart files:

- `oasis_oce.nc`, `oasis_wave.nc`: restart files for OASIS, you need to create them at the beginning of the run, OASIS will overwrite them at the end of the run, and they will be available for next restart
- `oce_ini`, `wave_ini`: correspond to `croco_ini.nc`, `restart.wv3`. These are your ocean and wave model initial or restart files

Practical example of the coupling sequence pictured in the above Figure:

```
oasis_time = 0
#1 => get field from oasis_wave.nc
rcv(0) => in oasis: get(0)
#2 => timestepping
t = 0+dt = 0+180 = 180
#3 => 180 is not a coupling time step, do nothing
snd(0) => in oasis: put(0+lag) = put(0+180) = put(180)
oasis_time = oasis_time+dt = 0+180 = 180
#4 => 180 is not a coupling time step, do nothing
rcv(180) => in oasis: get(180)
#5 => timestepping
t = 180+dt = 180+180 = 360
#6 => 360 is a coupling time step, put field
snd(180) => in oasis: put(180+lag) = put(180+180) = put(360)
```

16.1.4 Interpolations

The OASIS3-MCT coupler can process time transformations and 2D spatial interpolations of the exchanged fields. The 2D spatial interpolation, requested if models have different grids, is performed by the **scrip** library using SCRIPR keyword in the `namcouple`. Available interpolation types are:

BILINEAR	interpolation based on a local bilinear approximation
BICUBIC	interpolation based on a local bicubic approximation
CONSERV	1st or 2nd order conservative remapping
DISTWGT	distance weighted nearest-neighbour interpolation (N neighbours)
GAUSWGT	N nearest-neighbour interpolation weighted by their distance and a gaussian function

See OASIS manual for detailed informations.

Time transformations can also be performed by OASIS using `LOCTRANS` keyword in the `namcouple`. Available transformations are:

INSTANT	no time transformation, the instantaneous field is transfered
ACCUMUL	the field accumulated over the previous coupling period is exchanged
AVERAGE	the field averaged over the previous coupling period is transfered
T_MIN	the minimum value of the field for each source grid point over the previous coupling period is transfered
T_MAX	the maximum value of the field for each source grid point over the previous coupling period is transfered

16.2 Detailed OASIS implementation

16.2.1 In CROCO

The following routines are specifically built for coupling with OASIS and contain calls to OASIS intrinsic functions:

- `cpl_prism_init.F`: Manage the initialization phase of OASIS3-MCT : local MPI communicator
- `cpl_prism_define.F`: Manage the definition phase of OASIS3-MCT: domain partition, name of exchanged fields as read in the `namcouple`
- `cpl_prism_grid.F`: Manage the definition of grids for the coupler
- `cpl_prism_put.F`: Manage the sending of arrays from CROCO to the OASIS3-MCT coupler
- `cpl_prism_getvar.F`: Manage the generic reception from OASIS3-MCT.
- `cpl_prism_get.F`: Manage the specificity of each received variable: C-grid position, and field unit transformations

These routines are called in the the code in:

- `main.F`: Initialization, and finalization phases
- `get_initial.F`: Definition phase
- `zoom.F`: Initialization phase for AGRIF nested simulations
- `step.F`: Exchanges (sending and reception) of coupling variables

Other CROCO routines have also been slightly modified to introduce coupling:

- `testkeys.F`: To enable automatic linking to OASIS3-MCT libraries during compilation with `job-comp`
- `cppdefs.h`: Definition of the `OA_COUPLING` and `OW_COUPLING` cpp-keys, and the other related and requested cpp-keys, as `MPI`
- `set_global_definitions.h`: Definition of cpp-keys in case of coupling (`undef OPENMP`, `define MPI`, `define MPI_COMM_WORLD ocean_grid_comm: MPI_COMM_WORLD` generic MPI communicator is redefined as the local MPI communicator `ocean_grid_comm`, `undef BULK_FLUX`: no bulk OA parametrization)

- `mpi_roms.h`: Newly added to define variables related to OASIS3-MCT operations. It manage the MPI communicator, using either the generic `MPI_COMM_WORLD`, either the local MPI communicator created by OASIS3-MCT
- `read_inp.F`: Not reading atmospheric forcing files (`croco_frc.nc` and/or `croco_blk.nc`) in OA coupled mode

A schematic picture of the calls in CROCO is (with # name.F indicating the routine we enter in):

```
# main.F
if !defined AGRIF
call cpl_prism_init
else
call Agrif_MPI_Init
endif
...
call read_inp
...
call_get_initial
    # get_initial.F
    ...
    call cpl_prism_define
        # cpl_prism_define.F
        call prism_def_partition_proto
        call cpl_prism_grid
        call prism_def_var_proto
        call prism_enddef_proto
    oasis_time=0
# main.F
...
DO 1:NT
call step
    # step.F
    if ( (iif==-1).and.(oasis_time>=0).and.(nbsstep3d<ntimes) ) then
        call cpl_prism_get(oasis_time)
        # cpl_prism_get.F
        call cpl_prism_getvar
    endif
    call prestep3d
        call get_vbc
        ...
    call step2d
    ...
    call step3d_uv
    call step3d_t
    iif = -1
    nbsstep3d = nbsstep3d + 1
    if (iif==-1) then
        if (oasis_time>=0.and.(nbsstep3d<ntimes)) then
            call cpl_prism_put (oasis_time)
            oasis_time = oasis_time + dt
        endif
    endif
endif
# main.F
END DO
...
call prism_terminate_proto
...
```

16.2.2 In WW3

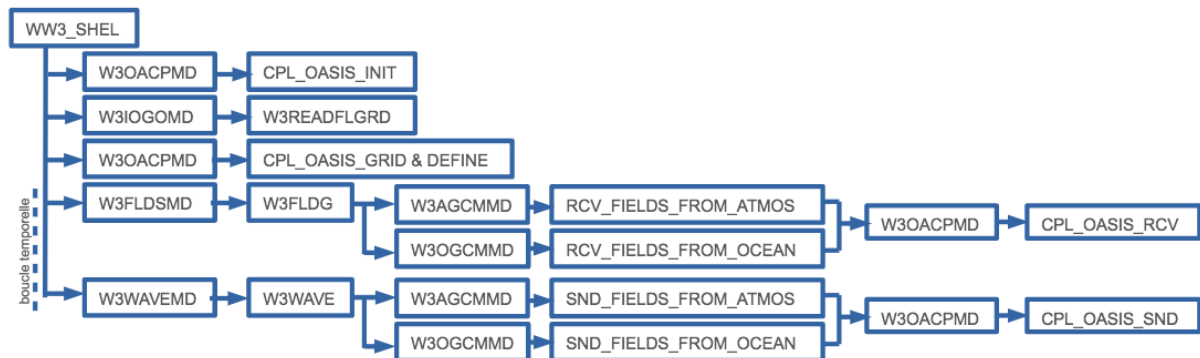
The following routines have been specifically built for coupling with OASIS:

- `w3oacpmd.ftn`: main coupling module with calls to oasis intrinsic functions
- `w3agcmmd.ftn`: module for coupling with an atmospheric model
- `w3ogcmmd.ftn`: module for coupling with an ocean model

The following routines have been modified for coupling with OASIS:

- `w3fldsmd.ftn`: routine that manage input fields, and therefore received fields from the coupler
- `w3wdatmd.ftn`: routine that manage data structure for wave model, and therefore time for coupling
- `w3wavemd.ftn`: actual wave model, here is located the sending of coupled variables
- `ww3_shel.ftn`: main routine managing the wave model, definition/initialisation/partition phases are located here

A schematic picture of the calls in WW3 is given here:



16.2.3 In WRF

The routines specifically built for coupling are:

- `module_cpl_oasis3.F`
- `module_cpl.F`

Implementation of coupling with the ocean implies modifications in the following routines:

- `phys/module_b1_mynn.F`
- `phys/module_b1_ySU.F`
- `phys/module_pbl_driver.F`
- `phys/module_surface_driver.F`
- `phys/module_sf_sfclay.F`
- `phys/module_sf_sfclayrev.F`

Implementation of coupling with waves implies modifications in the following routines:

- `Registry/Registry.EM_COMMON`: `CHA_COEF` added
- `dyn/module_first_rk_step_part1.F`: `CHA_COEF=grid%cha_coef` declaration added
- `frame/module_cpl.F`: `rcv CHA_COEF` added
- `phys/module_sf_sfclay.F` and `..._sfclayrev.F`: introduction of wave coupled case: `is-ftcflx=5` as follows:

```

! SJ: change charnock coefficient as a function of waves, and hence roughness
! length
IF ( ISFT CFLX.EQ.5 ) THEN
    ZNT(I)=CHA_COEF(I)*UST(I)*UST(I)/G+0.11*1.5E-5/UST(I)
ENDIF

```

- phys/module_surface_driver.F: CHA_COEF added in calls to sfclay and sfclayrev and “CALL cpl_rcv” for CHA_COEF

Schematic picture of WRF architecture and calls to the coupling dependencies:

```

# main/wrf.F
CALL wrf_init

# main/module_wrf_top.F
CALL wrf_dm_initialize

# frame/module_dm.F
CALL cpl_init( mpi_comm_here )
CALL cpl_abort( 'wrf_abort', 'look for abort message in rsl* files' )

CALL cpl_defdomain( head_grid )

# main/wrf.F
CALL wrf_run

# main/module_wrf_top.F
CALL integrate ( head_grid )

# frame/module_integrate.F
CALL cpl_defdomain( new_nest )
CALL solve_interface ( grid_ptr )

# share/solve_interface.F
CALL solve_em ( grid , config_flags ... )

# dyn_em/solve_em.F
curr_secs2 # time for the coupler
CALL cpl_store_input( grid, config_flags )
CALL cpl_settime( curr_secs2 )
CALL first_rk_step_part1

# dyn_em/module_first_rk_step_part1.F
CALL surface_driver( ... )

# phys/module_surface_driver.F
CALL cpl_rcv( id, ... )
u_phytmp(i,kts,j)=u_phytmp(i,kts,j)-uoce(i,j)
v_phytmp(i,kts,j)=v_phytmp(i,kts,j)-voce(i,j)

CALL SFCLAY( ... cha_coef ... )
# phys/module_sf_sfclay.F
CALL SFCLAY1D
IF ( ISFT CFLX.EQ.5 ) THEN
    ZNT(I)=CHA_COEF(I)*UST(I)*UST(I)/G+0.11*1.5E-5/UST(I)
ENDIF

CALL SFCLAYREV( ... cha_coef ... )
# phys/module_sf_sfclayrev.F
CALL SFCLAYREV1D
IF ( ISFT CFLX.EQ.5 ) THEN
    ZNT(I)=CHA_COEF(I)*UST(I)*UST(I)/G+0.11*1.5E-5/UST(I)

```

(continues on next page)

(continued from previous page)

```

ENDIF

# dyn_em/module_first_rk_step_part1.F
CALL pbl_driver( ... )

# phys/module_pbl_driver.F
CALL ysu( ... uoce,voce, ... )
# module_bl_ysu.F
call ysu2d ( ... uox,vox, ... )
wspd1(i) = sqrt( (ux(i,1)-uox(i))*(ux(i,1)-uox(i))
                + (vx(i,1)-vox(i))*(vx(i,1)-vox(i)) )+1.e-9
f1(i,1) = ux(i,1)+uox(i)*ust(i)**2*g/del(i,1)*dt2/wspd1(i)
f2(i,1) = vx(i,1)+vox(i)*ust(i)**2*g/del(i,1)*dt2/wspd1(i)

CALL mynn_bl_driver( ... uoce,voce, ... )
# module_bl_mynn.F
d(1)=u(k)+dtz(k)*uoce*ust**2/wspd
d(1)=v(k)+dtz(k)*voce*ust**2/wspd

# dyn_em/solve_em.F
CALL first_rk_step_part2

# frame/module_integrate.F
CALL cpl_snd( grid_ptr )

# Check where this routine is called...
# frame/module_io_quilt.F # for IO server (used with namelist variable: nio_
↪tasks_per_group
CALL cpl_set_dm_communicator( mpi_comm_local )
CALL cpl_finalize()

# main/wrf.F
CALL wrf_finalize
#main/module_wrf_top.F
CALL cpl_finalize()

```

16.3 Coupled variables

16.3.1 Coupling with an atmospheric model

When coupling CROCO to an atmospheric model, to have a consistent interface, you should use momentum and heat fluxes computed from the atmospheric model bulk formula.

No surface forcing file is required (only boundary forcing, and eventually tidal forcing).

The following cpp-keys have to be set:

```

# define OA_COUPLING
# define MPI

# undef BULK_FLUX
# undef SMFLUX_CFB

```

Note: SMFLUX_CFB is a cpp-key to use a wind stress relative to the current in forced mode. In coupled mode, as current is sent to the atmosphere, the wind stress from the atmospheric model account for such a current feedback.

Fields sent by CROCO		
Name (units)	name and eventual oper. in the model	OASIS name
SST (K)	$t(:, :, N, nnew, itemp) + 273.15$	SRMSSTV0
Zonal current (m/s)	u (at rho points): $0.5*(u(1:Lmmpi, 1:Mmmpi, N, nnew) + u(2:Lmmpi+1, 1:Mmmpi, N, nnew))$	SRMUOCE0
Meridional current (m/s)	v (at rho points): $0.5*(v(1:Lmmpi, 1:Mmmpi, N, nnew) + v(1:Lmmpi, 2:Mmmpi+1, N, nnew))$	SRMVOCE0

Fields received by CROCO		
Name (units)	name and eventual oper. in the model	OASIS name
Zonal wind stress (N/m ²)	sustr (at u point): $0.5*(FIELD(io-1, jo) + FIELD(io, jo)) / rho0$	RRMTAWX0
Meridional wind stress (N/m ²)	svstr (at v point): $0.5*(FIELD(io, jo-1) + FIELD(io, jo)) / rho0$	RRMTAWY0
Wind stress module (N/m ²)	$smstr = FIELD / rho0$	RRMTAUM0
Surface net solar flux (W/m ²)	$srfx = FIELD / (rho0 * Cp)$	RRMSRFL0
Surface net non-solar flux (W/m ²)	$stflx(:, :, itemp) = FIELD / (rho0 * Cp)$	RRMSTFL0
Evaporation-Precipitation (kg/m ² /s)	$stflx(:, :, isalt) = FIELD / 1000$	RRMEVPR0

Fields received by WRF		
Name (units)	name (in the model)	OASIS name
SST (K)	SST	WRF_d01_EXT_d01_SST
Zonal current (m/s)	UOCE	WRF_d01_EXT_d01_UOCE
Meridional current (m/s)	VOCE	WRF_d01_EXT_d01_VOCE

Fields sent by WRF		
Name (units)	name (in the model)	OASIS name
Surface net solar flux (W/m2)	GSW	WRF_d01_EXT_d01_SURF_NET_SOLAR
Surface net non-solar flux (W/m2)	GLW-STBOLT*EMISS*SST**4-LH-HFX	WRF_d01_EXT_d01_SURF_NET_NON-SOLAR
Evaporation-precipitation (kg/m2/s)	QFX-(RAINCV+RAINNCV)/DT	WRF_d01_EXT_d01_EVAP-PRECIP
Zonal wind stress (N/m2)	taut * u_uo / wspd	WRF_d01_EXT_d01_TAUZ
Meridional wind stress (N/m2)	taut * u_uo / wspd	WRF_d01_EXT_d01_TAUZ
Wind stress module (N/m2)	taut	WRF_d01_EXT_d01_TAUMOD

16.3.2 Coupling with a wave model

When coupling CROCO to a wave model, the wave-current interactions have to be set on. At the moment, only mean wave parameters are exchanged, their contribution to ocean dynamics is computed into the wave-current interaction routine in CROCO.

The following cpp-keys have to be set:

```
# define OW_COUPLING
# define MPI
# define MRL_WCI
```

Note: You also have to be careful to the choice of the momentum flux. For better consistency, here we suggest to account for the momentum flux seen by the wave model, and thus set:

```
# undef BULK_FLUX
# define WAVE_SMFLUX
```

Fields sent by CROCO		
Name (units)	name and eventual oper. in the model	OASIS name
SSH (m)	zeta	SRMSSHV0
Zonal current (m/s)	u (at rho points): 0.5*(u(1:Lmmpi,1:Mmmpi,N,nnew) +u(2:Lmmpi+1,1:Mmmpi,N,nnew))	SRMUOCE0
Meridional current (m/s)	v (at rho points): 0.5*(v(1:Lmmpi,1:Mmmpi,N,nnew) +v(1:Lmmpi,2:Mmmpi+1,N,nnew))	SRMVOCE0

Fields received by CROCO		
Name (units)	name and eventual oper. in the model	OASIS name
Significant wave height (m)	whrm = FIELD * 0.70710678	RRM__HS0
Mean wave period (s)	wfrq = 2*pi / FIELD	RRMT0M10
Cosine of mean wave direction	wdrx	RRMCDIR0
Sine of mean wave direction	wdre	RRMSDIR0
Zonal wave stress (m2/s2)	twox (at u point): 0.5*(FIELD(io-1,jo)+FIELD(io,jo))	RRMTWOX0
Meridional wave stress (m2/s2)	twoy (at v point): 0.5*(FIELD(io,jo-1)+FIELD(io,jo))	RRMTWOY0
Zonal wind stress (m2/s2)	tawx (at u point): 0.5*(FIELD(io-1,jo)+FIELD(io,jo))	RRMTAWX0
Meridional wind stress (m2/s2)	tawy (at v point): 0.5*(FIELD(io,jo-1)+FIELD(io,jo))	RRMTAWY0

Fields received by WW3		
Name (units)	name (in the model)	OASIS name
SSH = water level (m)	LEV	WW3__SSH
Zonal current (m/s)	CUR	WW3__OSSU
Meridional current (m/s)	CUR	WW3__OSSV

Fields sent by WW3		
Name (units)	name (in the model)	OASIS name
Mean wave period (s)	T0M1	WW3__T0M1
Significant wave height (m)	HS	WW3__OHS
Cosine of mean wave direction	cos(DIR)	WW3__CDIR
Sine of mean wave direction	sin(DIR)	WW3__SDIR
Zonal wave stress (N/m2)	TWOX	WW3__TWOX
Meridional wave stress (N/m2)	TWOY	WW3__TWOY
Zonal wind stress (N/m2)	TAWX	WW3__TAWX
Meridional wind stress(N/m2)	TAWY	WW3__TAWY
Other fields possibly sent, but not used in coupling with CROCO at the moment		
Bernoulli head pressure (N/m)	BHD	WW3__BHD
Bottom orbital velocity (m/s)	UBR	WW3__UBR
Wave-to-ocean TKE flux (W/m2)	FOC	WW3__FOC
Mean wavelength (m)	LM	WW3__LM
Wave peak frequency (/s)	FP	WW3__FP

16.3.3 Coupling atmosphere and wave models

Fields received by WW3		
Name (units)	name (in the model)	OASIS name
Zonal wind (m/s)	WND	WW3__U10
Meridional wind (m/s)	WND	WW3__V10

Fields sent by WW3		
Name (units)	name (in the model)	OASIS name
Significant wave height (m)	HS	WW3__AHS

Fields sent by WRF		
Name (units)	name (in the model)	OASIS name
Zonal wind at first level((m/s)	u_uo	WRF_d01_EXT_d01_U_01
Meridional wind at first level (m/s)	v_vo	WRF_d01_EXT_d01_V_01

Fields received by WRF		
Name (units)	name (in the model)	OASIS name
Charnock coefficient	CHA_COEF	WRF_d01_EXT_d01_CHA_COEF

16.3.4 Note on momentum flux when coupling 3 models

As the wave model has a quite complex parameterization of wave generation by winds, which is in subtle balance with the wave dissipation, the wind stress for the wave model is computed by its own parameterization. Therefore, to ensure energetic consistency of the momentum flux when coupling 3 models, we prescribe the wind stress in CROCO as:

```

sustr = sustr_from_atm_model - tawx + twox
svstr = svstr_from_atm_model - tawy + twoy

# where taw is stress from atm to waves
# and two is stress from waves to ocean
    
```

16.4 Grids

16.4.1 OASIS grid files

OASIS manage grids and interpolations by using dedicated grid files:

- `grids.nc`
- `masks.nc`
- `areas.nc` (requested only for some of the interpolation types)

These files can be automatically created by OASIS functions called in each model, or can be created by the user in advance if specificities are requested. Some facilities are provided in `croco_tools/Coupling_tools` to create such grids.

If `grids.nc`, `masks.nc`, `areas.nc` exist in the working directory, they won't be overwritten by OASIS functions. So, be sure to have the good files or remove them before running the coupled model.

16.4.2 Multiple model grids (nesting case)

Multiple nested grids in the different models can be used in coupled mode.

The variables are therefore exchanged from/to the different grids. To do so, each coupled variable is identified in the coupler with its grid number:

- For CROCO the last character of the OASIS variable name defines the domain
 - 0 being the parent domain
 - 1 the first child domain, etc.
- For WRF the domains are defined by d01, d02, etc, and the target domain (CROCO for instance), by EXT_d01, EXT_d02, etc.

For example if you are coupling 2 CROCO domains to one atmospheric domain, you will specify 2 types of exchanges in the `namcouple`:

```
# exchange between CROCO parent domain to WRF domain
SRMSSTV0 WRF_d01_EXT_d01_SST

# exchange between CROCO child domain to WRF domain
SRMSSTV1 WRF_d01_EXT_d02_SST
```

If you are coupling 2 WRF domains to one CROCO domain:

```
# exchange between WRF d01 domain to CROCO domain
WRF_d01_EXT_d01_TAUMOD RRMTAUM0

# exchange between WRF d02 domain to CROCO domain
WRF_d02_EXT_d01_TAUMOD RRMTAUM0
```

Related CPP options:

OW_COUPLING	Activate Ocean-Wave coupling
OA_COUPLING	Activate Ocean-Atmosphere coupling
OA_MCT	Use OASIS-MCT for coupling

Preselected options:

```
#undef OA_COUPLING
#undef OW_COUPLING
```

I/O AND ONLINE DIAGNOSTICS

Related CPP options:

AVERAGES	Process and output time-averaged data
AVERAGES_K	Process and output time-averaged vertical mixing
DIAGNOSTICS_TS	Store and output budget terms of the tracer equations
DIAGNOSTICS_TS_ADV	Choose advection rather than transport formulation for tracer budgets
DIAGNOSTICS_TS_MLD	Integrate tracer budgets over the mixed-layer depth
DIAGNOSTICS_UV	Store and output budget terms of the momentum equations
XIOS and XIOS2	Use XIOS IO server (only version ≥ 2 is supported)

XIOS is an external library for output (developed at IPSL) providing for flexibility and design to improve performances for HPC : see <http://forge.ipsl.jussieu.fr/ioserver>

Preselected options:

```
# define AVERAGES
# define AVERAGES_K
# undef DIAGNOSTICS_TS
# undef DIAGNOSTICS_UV
# ifdef DIAGNOSTICS_TS
# undef DIAGNOSTICS_TS_ADV
# undef DIAGNOSTICS_TS_MLD
# endif
# undef XIOS
```


REVIEW OF TEST CASES

18.1 Basin

This is a rectangular, flat-bottomed basin with double-gyre wind forcing. It produces a western boundary current flowing into a central Gulf Stream which goes unstable and generates eddies if resolution is increased

```
# define BASIN
```

CPP options:

```
# undef OPENMP
# undef MPI
# define UV_ADV
# define UV_COR
# define UV_VIS2
# define SOLVE3D
# define TS_DIF2
# define BODYFORCE
# define ANA_GRID
# define ANA_INITIAL
# define ANA_SMFLUX
# define ANA_STFLUX
# define ANA_BTFLUX
# define NO_FRCFIELD
```

Settings :

Results :

18.2 Canyon

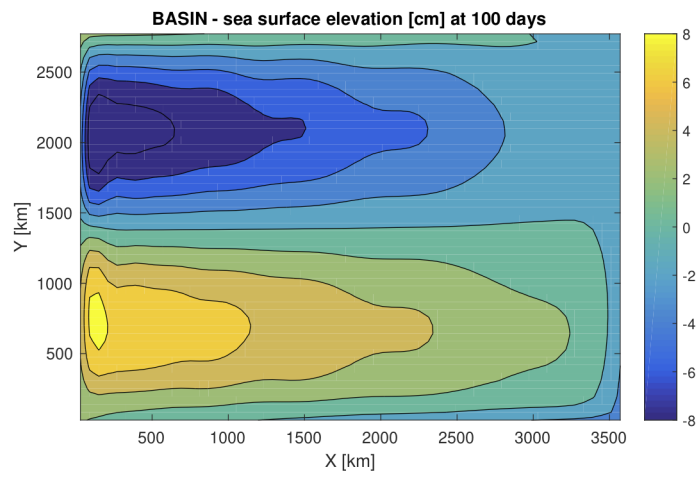
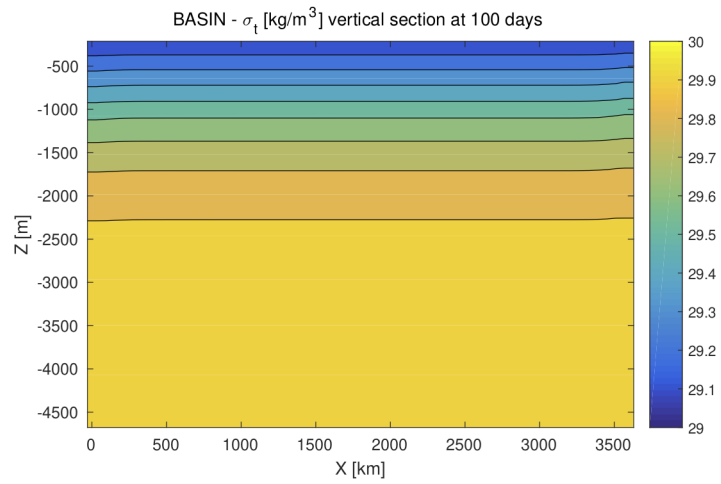
```
# define CANYON
```

CPP options:

```
# undef OPENMP
# undef MPI
# define CANYON_STRAT
# define UV_ADV
# define UV_COR
# define SOLVE3D
# define EW_PERIODIC
# define ANA_GRID
# define ANA_INITIAL
# define ANA_SMFLUX
# define ANA_STFLUX
```

(continues on next page)

BASIN



tags/v1.1_beta-0-g5d3de5f7

DATE: 08-10-2019 TIME: 18:03:43

Fig. 1: BASIN results : density (up) and sea surface elevation (down)

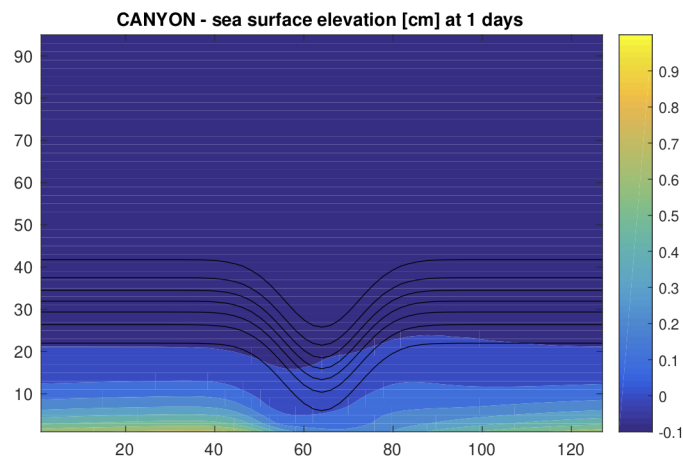
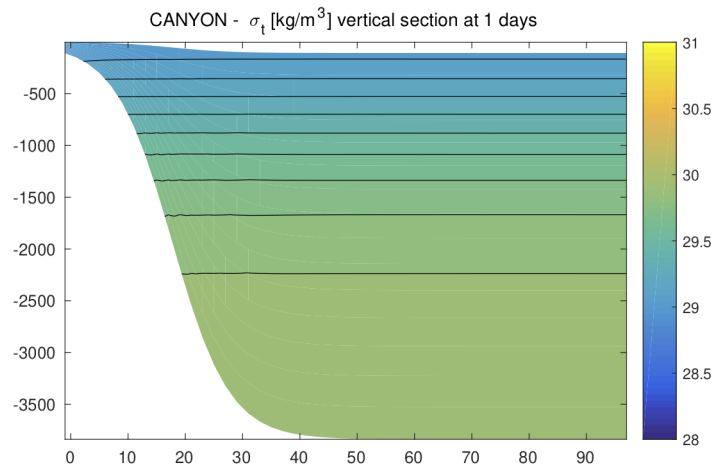
(continued from previous page)

```
# define ANA_BTFLUX
# define NO_FRCFILE
```

Settings :

Results :

CANYON



tags/v1.1_beta-0-g5d3de5f7

DATE: 08-10-2019 TIME: 18:03:59

Fig. 2: CANYON results : density (up) and sea surface elevation (down)

18.3 Equator

Boccaletti, G., R.C. Pacanowski, G.H. Philander and A.V. Fedorov, 2004, The Thermal Structure of the Upper Ocean, J.Phys.Oceanogr., 34, 888-902.

```
# define EQUATOR
```

CPP options:

```
# undef OPENMP
# undef MPI
# define UV_ADV
# define UV_COR
# define UV_VIS2
# define SOLVE3D
# define TS_DIF2
# define ANA_GRID
# define ANA_INITIAL
# define ANA_SMFLUX
# define ANA_STFLUX
# define ANA_SRFLUX
# define ANA_SSFLUX
# define ANA_BTFLUX
# define ANA_BSFLUX
# define QCORRECTION
# define ANA_SST
# define LMD_MIXING
# define LMD_RIMIX
# define LMD_CONVEC
# define NO_FRCTILE
```

Settings :

Results :

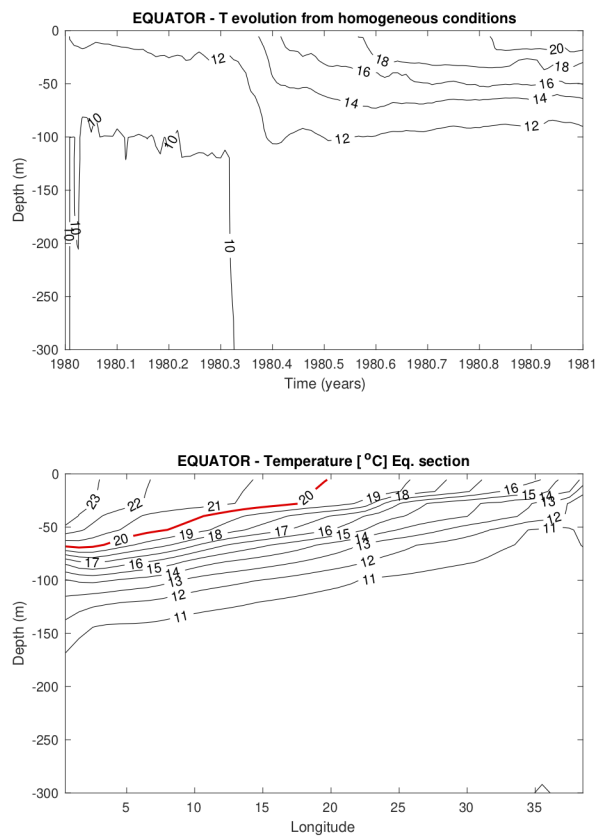


Fig. 3: EQUATOR results : temperature , time evolution (up) vertical section (down)

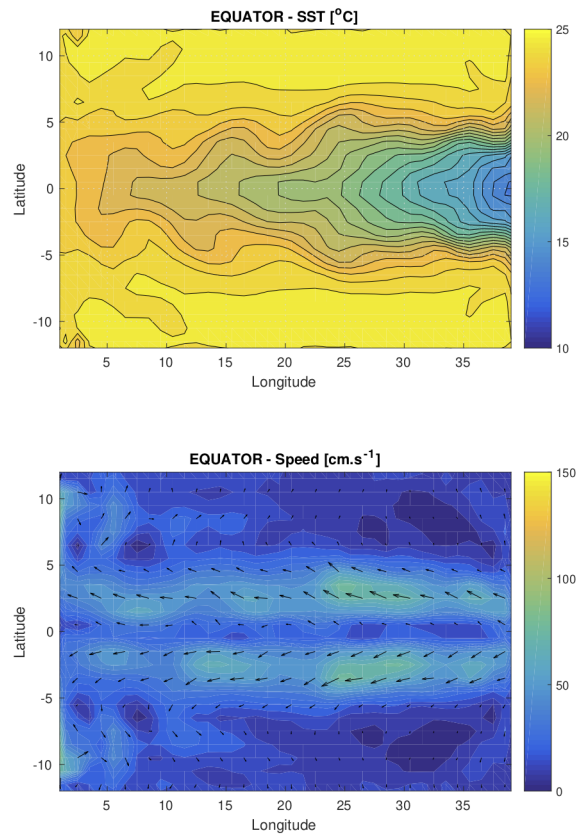


Fig. 4: EQUATOR results : speed (up) and sea surface elevation (down)

18.4 Inner Shelf

Compare wind driven innershelf dynamics between 2D Ekman theory and CROCO numerical solution

Estrade P., P. Marchesiello, A. Colin de Verdiere, C. Roy, 2008: Cross-shelf structure of coastal upwelling : a two-dimensional expansion of Ekman's theory and a mechanism for innershelf upwelling shut down. *Journal of Marine Research*, 66, 589-616.

Marchesiello P., and P. Estrade, 2010: Upwelling limitation by geostrophic onshore flow. *Journal of Marine Research*, 68, 37-62.

```
# define INNERSHELF
```

CPP options:

```
# undef OPENMP
# undef MPI
# undef NBQ
# define INNERSHELF_EKMAN
# define INNERSHELF_APG
# define SOLVE3D
# define UV_COR
# define ANA_GRID
# define ANA_INITIAL
# define AVERAGES
# define ANA_SSFLUX
# define ANA_SRFLUX
# define ANA_STFLUX
# define ANA_BSFLUX
# define ANA_BTFLUX
```

(continues on next page)

```
# define ANA_SMFLUX
# define NS_PERIODIC
# define OBC_WEST
# define SPONGE
# ifndef INNERSHELF_EKMAN
# define UV_ADV
# define SALINITY
# define NONLIN_EOS
# define LMD_MIXING
# undef GLS_MIXING
# ifdef LMD_MIXING
# define LMD_SKPP
# define LMD_BKPP
# define LMD_RIMIX
# define LMD_CONVEC
# endif
# undef WAVE_MAKER_INTERNAL
# ifdef WAVE_MAKER_INTERNAL
# define ANA_BRY
# define Z_FRC_BRY
# define M2_FRC_BRY
# define M3_FRC_BRY
# define T_FRC_BRY
# endif
# endif
# define NO_FRCFILE
```

Settings :

Results :

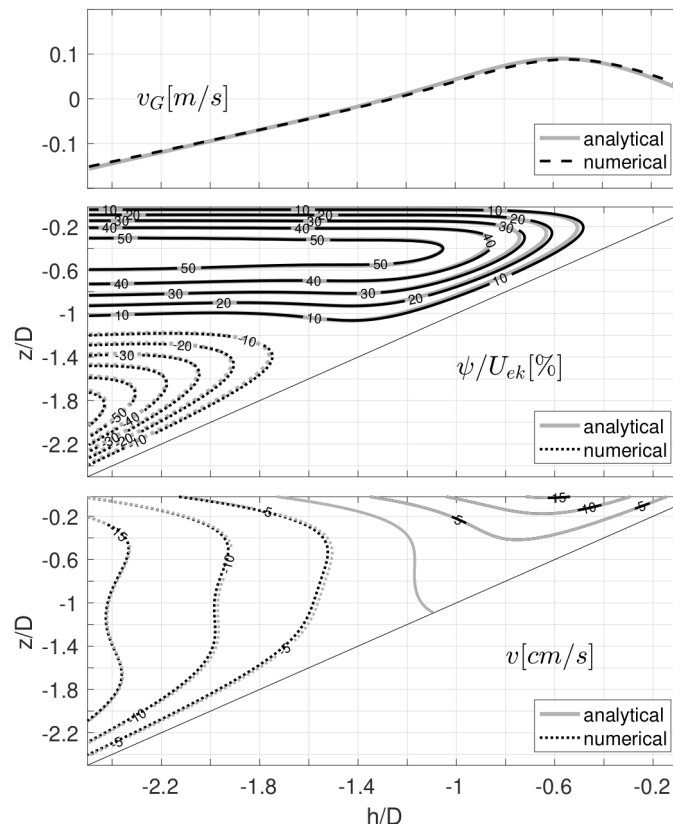


Fig. 5: INNERSHELF results : comparison with analytical solution

18.5 River Runoff

```
# define RIVER
```

CPP options:

```
# undef OPENMP
# undef MPI
# define SOLVE3D
# define UV_ADV
# define UV_COR
# define NONLIN_EOS
# define SALINITY
# define ANA_GRID
# define MASKING
# define ANA_INITIAL
# define ANA_SMFLUX
# define ANA_STFLUX
# define ANA_SSFLUX
# define ANA_SRFLUX
# define ANA_BTFLUX
# define ANA_BSFLUX
# define LMD_MIXING
# define LMD_SKPP
# define LMD_BKPP
# define LMD_RIMIX
# define LMD_CONVEC
# define PSOURCE
# define ANA_PSOURCE
# define NS_PERIODIC
# undef FLOATS
# ifdef FLOATS
#   define RANDOM_WALK
#   ifdef RANDOM_WALK
#     define DIEL_MIGRATION
#     define RANDOM_VERTICAL
#     define RANDOM_HORIZONTAL
#   endif
# endif
# define NO_FRCFILE
```

Setting :

Results :

18.6 Plume

```
# define PLUME
```

CPP options:

```
#elif defined PLUME
# undef OPENMP
# undef MPI
# define NEW_S_COORD
# define ANA_GRID
# define ANA_INITIAL
# define ANA_SMFLUX
# define ANA_STFLUX
```

(continues on next page)

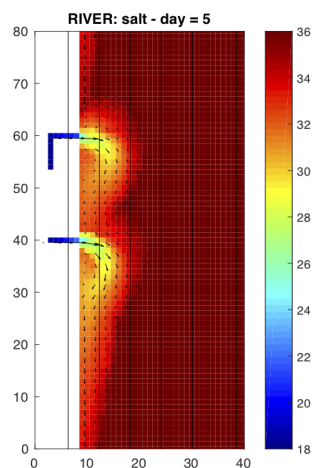


Fig. 6: RIVER results : river plume

(continued from previous page)

```
# define ANA_SSFLUX
# define ANA_SRFLUX
# define ANA_BSFLUX
# define ANA_BTFLUX
# define SOLVE3D
# define UV_COR
# define UV_ADV
# define SALINITY
# define NONLIN_EOS
# define SPLIT_EOS
# define TS_HADV_UP3
# define SPONGE
# undef LMD_MIXING
# define GLS_MIXING
# ifdef LMD_MIXING
# define LMD_SKPP
# define LMD_BKPP
# define LMD_RIMIX
# define LMD_CONVEC
# undef LMD_DDMIX
# define LMD_NONLOCAL
# undef MLCONVEC
# endif
# define NO_FRCFILE
```

18.7 Gravitational/Overflow

```
# define OVERFLOW
```

CPP options:

```
# undef OPENMP
# undef MPI
# define UV_ADV
# define UV_COR
# define UV_VIS2
# define TS_DIF2
# define TS_MIX_GEO
# define SOLVE3D
```

(continues on next page)

(continued from previous page)

```
# define ANA_GRID
# define ANA_INITIAL
# define ANA_SMFLUX
# define ANA_STFLUX
# define ANA_BTFLUX
# define NO_FRCFILE
```

Setting :

Results :

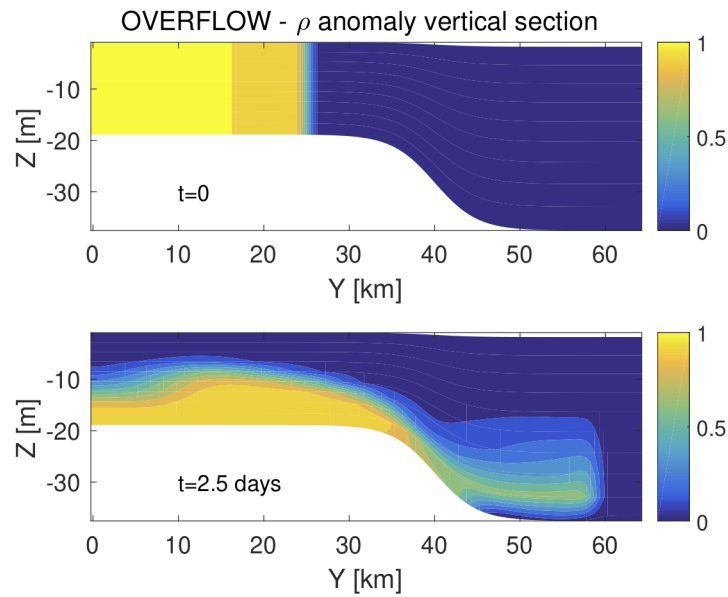


Fig. 7: OVERFLOW results : initial state (up) and density evolution (down)

18.8 Seamount

```
# define SEAMOUNT
```

CPP options:

```
# undef OPENMP
# undef MPI
# define UV_ADV
# define UV_COR
# define SOLVE3D
# define SALINITY
# define NONLIN_EOS
# define ANA_GRID
# define ANA_INITIAL
# define ANA_SMFLUX
# define ANA_STFLUX
# define ANA_SSFLUX
# define ANA_SRFLUX
# define ANA_BTFLUX
# define ANA_BSFLUX
# define NO_FRCFILE
```

Settings :

Results :

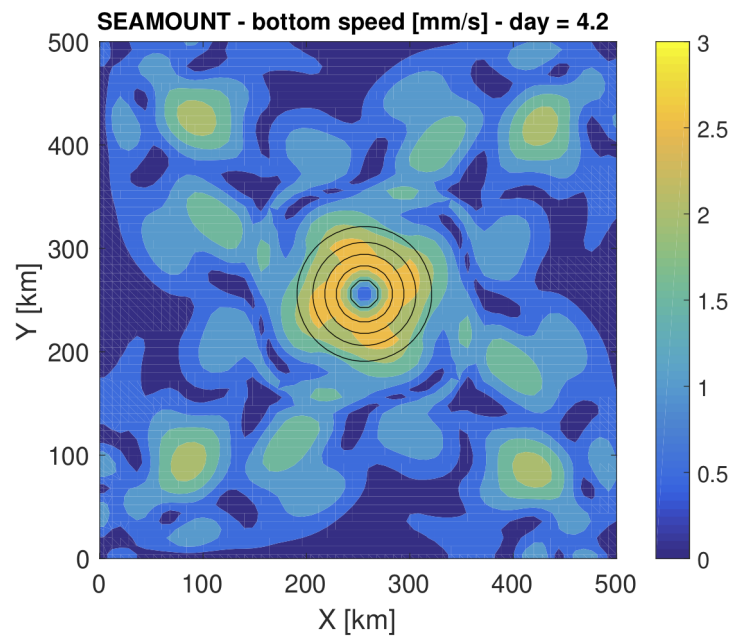


Fig. 8: SEAMOUNT results : bottom speed

18.9 Shelf front

```
# define SHELF FRONT
```

CPP options:

```
# undef OPENMP
# undef MPI
# define UV_ADV
# define UV_COR
# define SOLVE3D
# define SALINITY
# define ANA_GRID
# define ANA_INITIAL
# define ANA_SMFLUX
# define ANA_STFLUX
# define ANA_SSFLUX
# define ANA_SRFLUX
# define ANA_BTFLUX
# define ANA_BSFLUX
# define EW_PERIODIC
# define NO_FRCFILE
```

Settings :

Results :

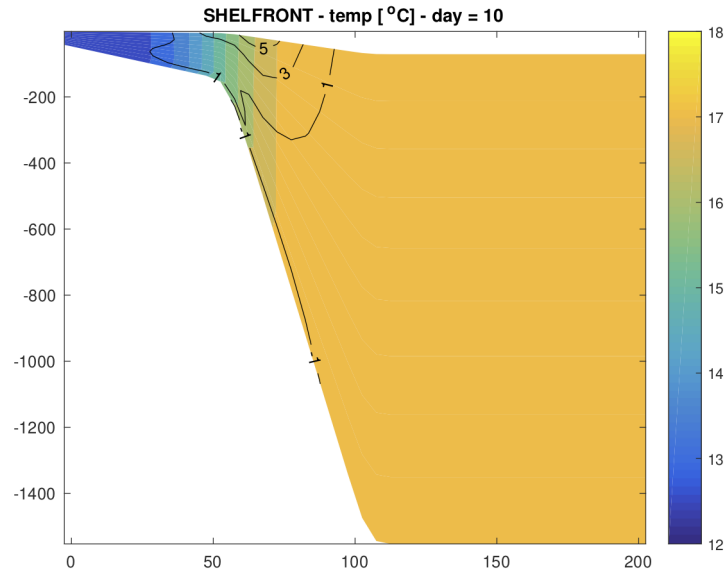


Fig. 9: SHELFROnt results : temperature profile

18.10 Equatorial Rossby Wave

This test problem considers the propagation of a Rossby soliton on an equatorial beta-plane, for which an asymptotic solution exists to the inviscid, nonlinear shallow water equations. In principle, the soliton should propagate westwards at fixed phase speed, without change of shape. Since the uniform propagation and shape preservation of the soliton are achieved through a delicate balance between linear wave dynamics and nonlinearity, this is a good context in which to look for erroneous wave dispersion and/or numerical damping.

The problem is nondimensionalized with: $H = 40$ cm, $L = 295$ km, $T = 1.71$ days and $U = L/T = 1.981$ m/s. Theoretical propagation speed is 0.4 (0.395) so that at $t = 120$, the soliton should be back to its initial position after crossing the periodic channel of length 48.

Boyd J.P., 1980: Equatorial solitary waves. Part I: Rossby solitons. JPO, 10, 1699-1717

```
# define SOLITON
```

CPP options:

```
# undef OPENMP
# undef MPI
# define UV_COR
# define UV_ADV
# define ANA_GRID
# define ANA_INITIAL
# define AVERAGES
# define EW_PERIODIC
# define ANA_SMFLUX
# define NO_FRCFIELD
```

Settings :

Results :

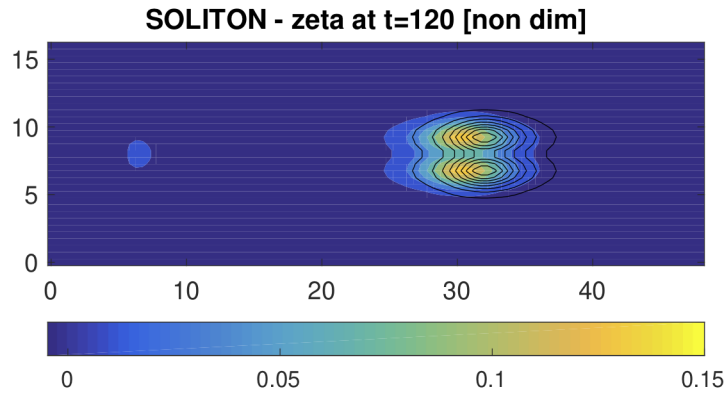


Fig. 10: SOLITON results : sea surface evolution

18.11 Thacker

Thacker, W., (1981), Some exact solutions to the nonlinear shallow-water wave equations. J. Fluid Mech., 107.

```
# define THACKER
```

CPP options:

```
# undef OPENMP
# undef MPI
# define THACKER_2DV
# define SOLVE3D
# define UV_COR
# define UV_ADV
# undef UV_VIS2
# define WET_DRY
# define NEW_S_COORD
# define ANA_GRID
# define ANA_INITIAL
# define ANA_BTFLUX
# define ANA_SMFLUX
# define ANA_SRFLUX
# define ANA_STFLUX
# define NO_FRCFIELD
```

Settings :

Results :

18.12 Upwelling

```
# define UPWELLING
```

CPP options:

```
# undef OPENMP
# undef MPI
# define SOLVE3D
# define UV_COR
# define UV_ADV
# define ANA_GRID
```

(continues on next page)

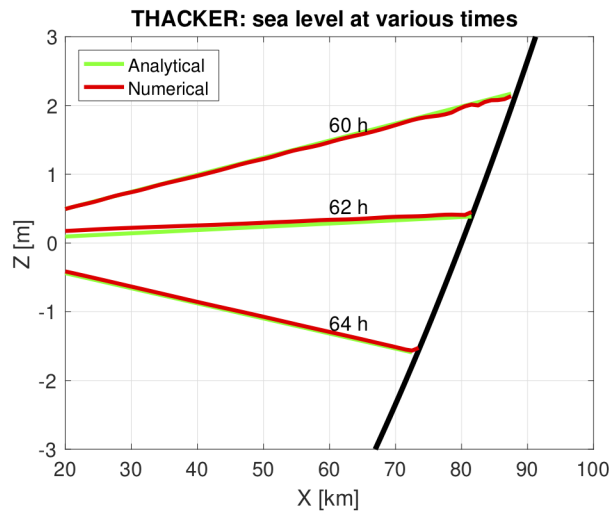


Fig. 11: THACKER results : elevation

(continued from previous page)

```

# define ANA_INITIAL
# define AVERAGES
# define SALINITY
# define NONLIN_EOS
# define ANA_SSFLUX
# define ANA_SRFLUX
# define ANA_STFLUX
# define ANA_BSFLUX
# define ANA_BTFLUX
# define ANA_SMFLUX
# define LMD_MIXING
# define LMD_SKPP
# define LMD_BKPP
# define LMD_RIMIX
# define LMD_CONVEC
# define EW_PERIODIC
# define NO_FRCFIELD
    
```

Settings :

Results :

18.13 Baroclinic Vortex

Free evolution of a baroclinic vortex (South West drift) that retains part of its initial axisymmetric shape as advective effects compensate for weak-amplitude Rossby-wave dispersion in its wake. 1-way and 2-way nesting were tested with this configuration.

McWilliams, J.C., Flierl, G.R., 1979. On the evolution of isolated nonlinear vortices. *J. Phys. Oceanogr.* 9, 1155–1182.

Penven, P., Debreu, L., Marchesiello, P., McWilliams, J.C., 2006. Application of the ROMS embedding procedure for the central California upwelling system. *Ocean Model.* 12, 157–187.

Debreu L., P. Marchesiello, P. Penven and G. Cambon, 2012: Two-way nesting in split-explicit ocean models: Algorithms, implementation and validation. *Ocean Modelling* 49-50 (2012) 1–21

```

# define VORTEX
    
```

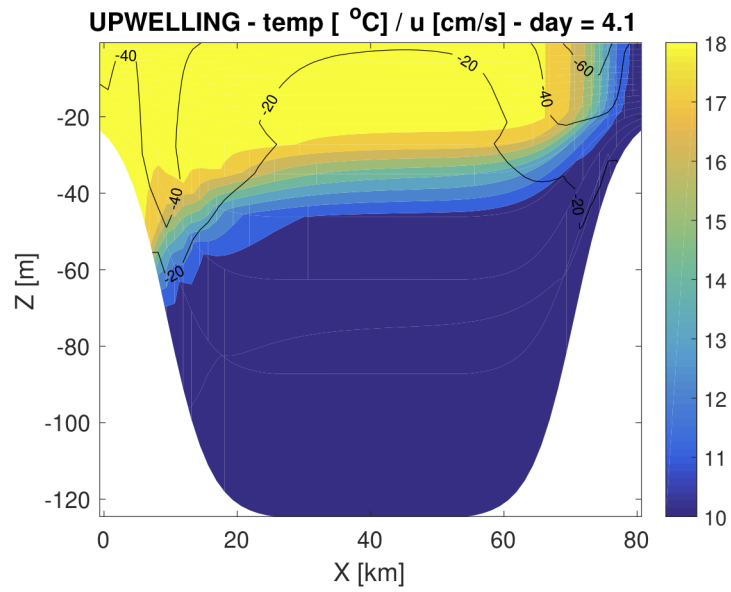


Fig. 12: UPWELLING results : temperature

CPP options:

```
# undef OPENMP
# undef MPI
# undef AGRIF
# undef AGRIF_2WAY
# undef NBQ
# define SOLVE3D
# define UV_COR
# define UV_ADV
# define ANA_STFLUX
# define ANA_SMFLUX
# define ANA_BSFLUX
# define ANA_BTFLUX
# define ANA_VMIX
# define OBC_EAST
# define OBC_WEST
# define OBC_NORTH
# define OBC_SOUTH
# define SPONGE
# define ZCLIMATOLOGY
# define M2CLIMATOLOGY
# define M3CLIMATOLOGY
# define TCLIMATOLOGY
# define ZNUDGING
# define M2NUDGING
# define M3NUDGING
# define TNUDGING
# define NO_FRCFILE
```

Settings :

Results :

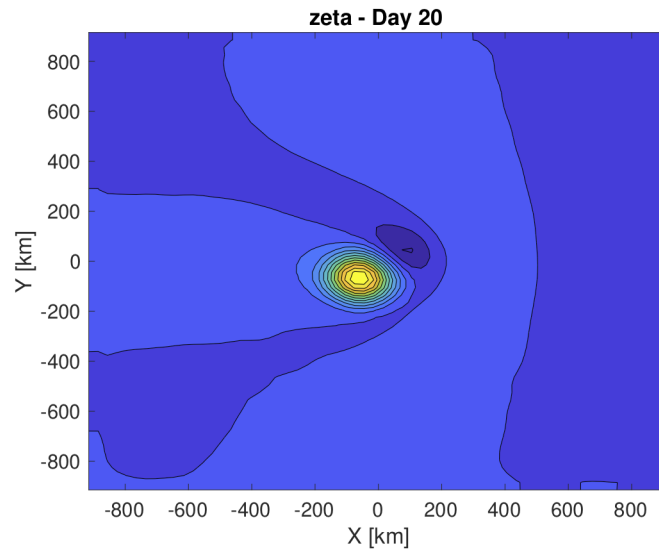


Fig. 13: VORTEX results : difference between parent and child grid (cm)

18.14 Internal Tide

Internal Gravity Wave solution over a ridge.

Di Lorenzo, E, W.R. Young and S.L. Smith, 2006, Numerical and analytical estimates of M2 tidal conversion at steep oceanic ridges, *J. Phys. Oceanogr.*, 36, 1072-1084.

```
# define INTERNAL
```

CPP options:

```
# undef OPENMP
# undef MPI
# define SOLVE3D
# define UV_COR
# define UV_ADV
# define BODYTIDE
# define ANA_GRID
# define ANA_INITIAL
# define ANA_BTFLUX
# define ANA_SMFLUX
# define ANA_SRFLUX
# define ANA_STFLUX
# define ANA_VMIX
# define EW_PERIODIC
# define NS_PERIODIC
# undef INTERNALSHELF
# ifdef INTERNALSHELF
# undef EW_PERIODIC
# define OBC_EAST
# define OBC_WEST
# define SPONGE
# define ANA_SSH
# define ANA_M2CLIMA
# define ANA_M3CLIMA
# define ANA_TCLIMA
# define ZCLIMATOLOGY
# define M2CLIMATOLOGY
# define M3CLIMATOLOGY
```

(continues on next page)

(continued from previous page)

```
# define TCLIMATOLOGY
# define M2NUDGING
# define M3NUDGING
# define TNUDGING
# endif
# define NO_FRCFILE
```

Setting :

Results :

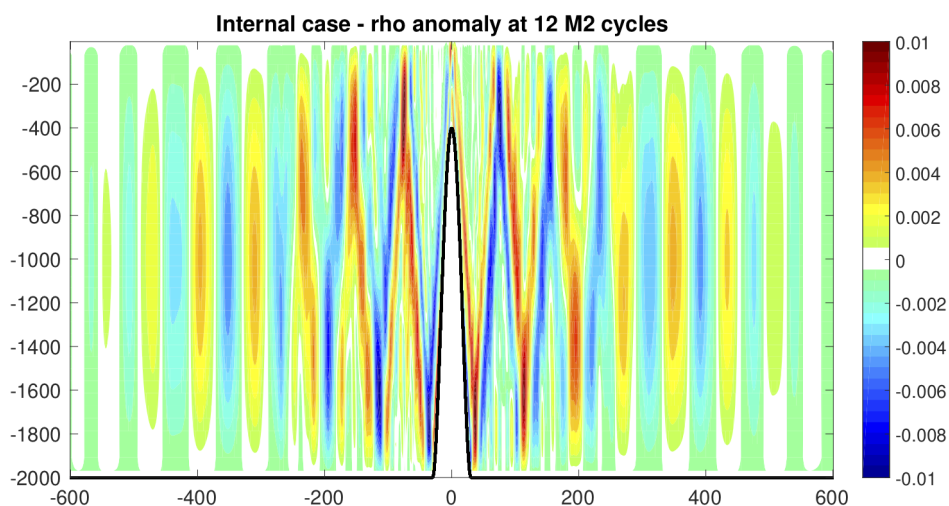


Fig. 14: INTERNAL results : generation of an internal wave

18.15 Internal Tide (COMODO)

Internal Gravity Wave solution over continental slope and shelf (COMODO test)

Pichon, A., 2007: Tests academiques de maree, Rapport interne n 21 du 19 octobre 2007, Service Hydrographique et Oceanographique de la Marine.

```
# define IGW
```

CPP options:

```
# define EXPERIMENT3
# undef OPENMP
# undef MPI
# undef NBQ
# define NEW_S_COORD
# define TIDES
# define TIDERAMP
# define SSH_TIDES
# define UV_TIDES
# define SOLVE3D
# define UV_ADV
# define UV_COR
# define UV_VIS2
# undef VADV_ADAPT_IMP
# define SPHERICAL
# define CURVGRID
# define ANA_INITIAL
```

(continues on next page)

(continued from previous page)

```
# define ANA_VMIX
# define ANA_SMFLUX
# define ANA_STFLUX
# define ANA_SRFLUX
# define ANA_SSFLUX
# define ANA_BTFLUX
# define ANA_BSFLUX
# define NS_PERIODIC
# define OBC_EAST
# define OBC_WEST
# undef SPONGE
# define ANA_SSH
# define ANA_M2CLIMA
# define ANA_M3CLIMA
# define ANA_TCLIMA
# define ZCLIMATOLOGY
# define M2CLIMATOLOGY
# define M3CLIMATOLOGY
# define TCLIMATOLOGY
# define M2NUDGING
# define M3NUDGING
# define TNUDGING
# undef ONLINE_ANALYSIS
```

Settings :

Settings :

Results :

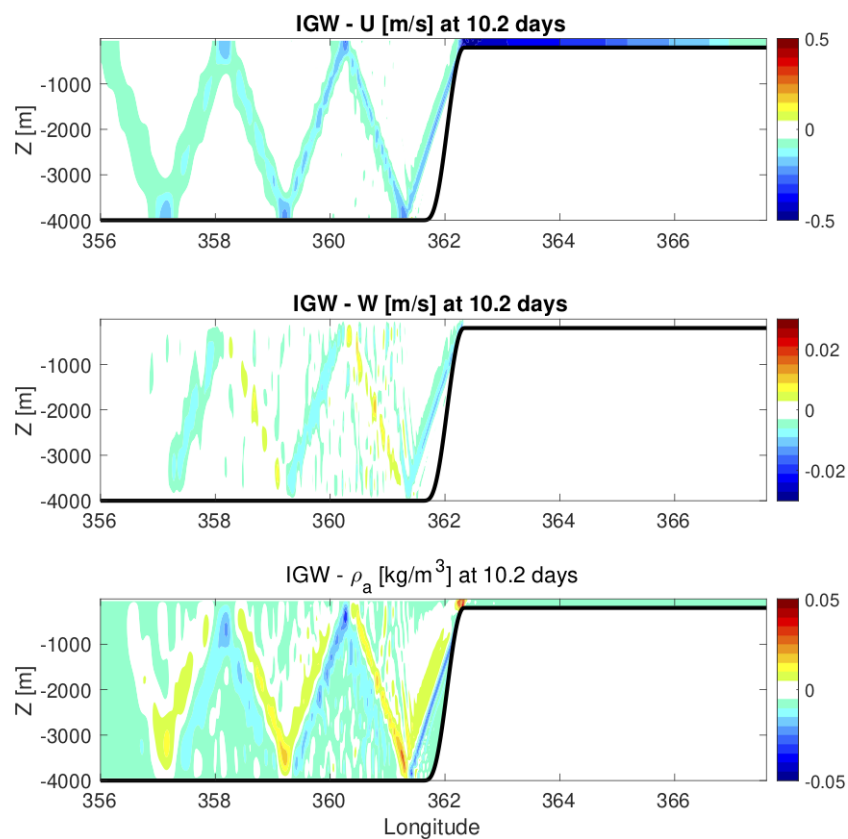


Fig. 15: IGW results : internal gravity waves generation

18.16 Baroclinic Jet

Effective resolution is limited by the numerical dissipation range, which is a function of the model numerical filters (assuming that dispersive numerical modes are efficiently removed). Soufflet et al. (2016) present a Baroclinic jet test case set in a zonally reentrant channel that provides a controllable test of a model capacity at resolving submesoscale dynamics.

A semi-idealized configuration in a periodic channel is set up to generate two dominant mechanisms of upper ocean turbulence: (i) surface density stirring by mesoscale eddies and (ii) fine scale instabilities directly energizing the submesoscale range. The setup consists of a flat reentrant channel of 500 km by 2000 km by 4000 m, centered around 30 deg of latitude on a β -plane (the Coriolis frequency is $1.10^{-4} s^{-1}$ at the center, $\beta = 1.610^{-11} m^{-1} s^{-1}$). Eastern/western boundary conditions are periodic while northern/southern conditions are closed. The initial density field is constructed with interior and surface meridional density gradients and associated geostrophic currents that are linearly unstable to both interior baroclinic and Charney instability modes. A linear stability analysis provides the exponential growth rate of unstable modes as a function of wavenumber. The two most unstable modes are clearly distinct in length scales on either side of the Rossby deformation radius (around 30 km in the center +/- 5 km from south to north). The interior geostrophic instability thus injects energy at mesoscale and Charney instability at submesoscale if resolution allows (2 km). The default resolution is 20 km (40 vertical levels) where only mesoscale instabilities are at work.

Soufflet Y., Marchesiello P., Lemarie F., Jouanno Julien, Capet X., Debreu L., Benshila R. (2016). On effective resolution in ocean models. *Ocean Modelling*, 98, 36-50.

```
# define JET
```

CPP options:

```
# define ANA_JET
# undef MPI
# undef NBQ
# define SOLVE3D
# define UV_COR
# define UV_ADV
# define UV_VIS2
# ifdef ANA_JET
# define ANA_GRID
# define ANA_INITIAL
# endif
# define ANA_STFLUX
# define ANA_SMFLUX
# define ANA_BSFLUX
# define ANA_BTFLUX
# define ANA_VMIX
# define EW_PERIODIC
# define CLIMATOLOGY
# ifdef CLIMATOLOGY
# define ZCLIMATOLOGY
# define M2CLIMATOLOGY
# define M3CLIMATOLOGY
# define TCLIMATOLOGY
# define ZNUDGING
# define M2NUDGING
# define M3NUDGING
# define TNUDGING
# define ROBUST_DIAG
# define ZONAL_NUDGING
# ifdef ANA_JET
# define ANA_SSH
# define ANA_M2CLIMA
# define ANA_M3CLIMA
# define ANA_TCLIMA
```

(continues on next page)

(continued from previous page)

```
# endif
# endif
# define LMD_MIXING
# ifdef LMD_MIXING
# undef ANA_VMIX
# define ANA_SRFLUX
# undef LMD_KPP
# define LMD_RIMIX
# define LMD_CONVEC
# endif
# define NO_FRCFILE
```

Setting :

Results :

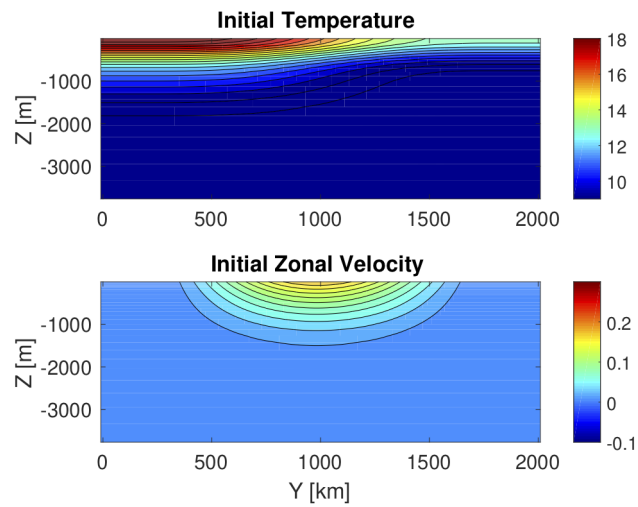


Fig. 16: JET results : initial state

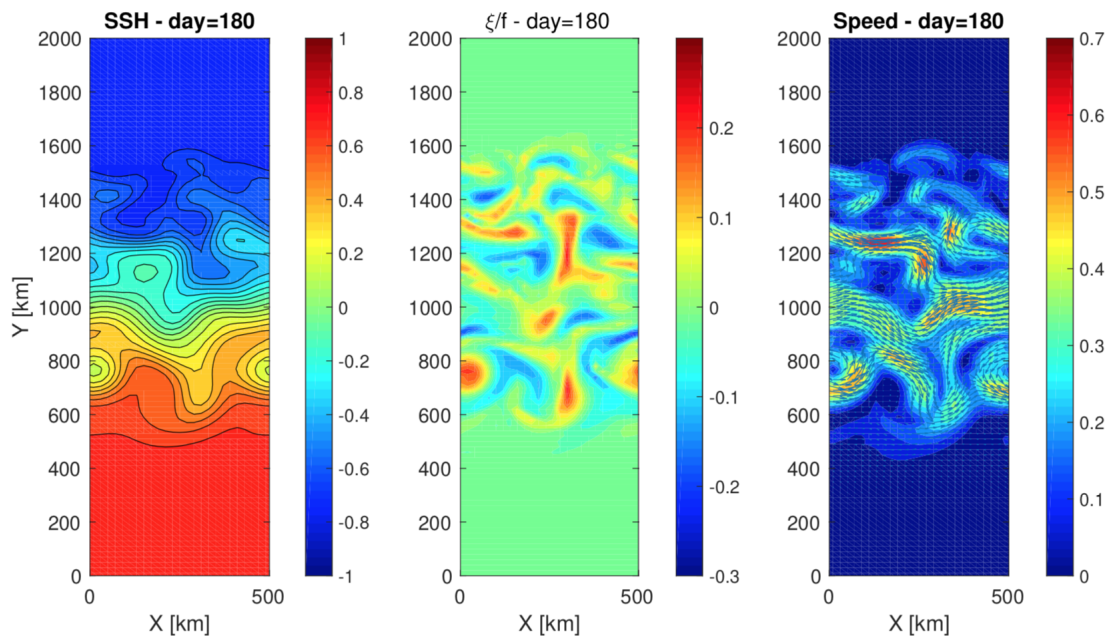


Fig. 17: JET results : results after 180 days

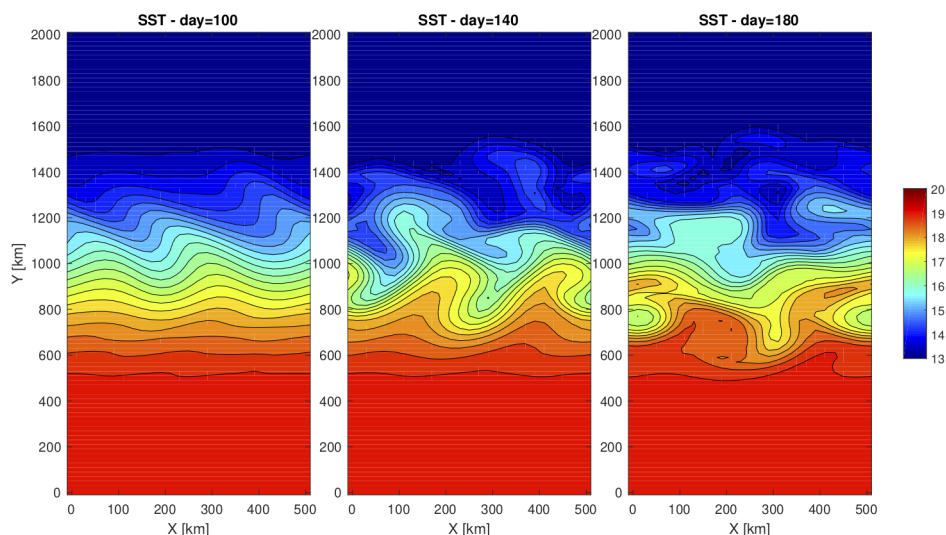


Fig. 18: JET results : vorticity evolution

18.17 Plannar Beach

This test case is a littoral flow driven by obliquely incident waves on a plane beach with a uniform slope of 1:80. The model is forced by monochromatic waves computed with the WKB wave model (Uchiyama et al., 2010) propagating offshore waves with 2 m significant wave height, a peak period of 10 s at an angle of 10° off the shore-normal direction. The horizontal extent of the domain is 1180 m in x (cross- shore), 140 m in y (alongshore) with grid spacings of $dx = dy = 20$ m. The model coordinates have a west-coast orientation, with the offshore open boundary located at $x = 0$. The resting depth h varies linearly from 12 m offshore, and is discretized with 20 uniform vertical sigma levels. Boundary conditions are alongshore periodicity, wetting-drying conditions at shore and open boundary conditions at the offshore boundary. Rotation is excluded with $f = 0$. There is no lateral momentum diffusion, stratification, nor surface wind/heat/freshwater fluxes. Breaking acceleration is given by the Church and Thornton (1993) formulation in the WKB model and wave-enhanced vertical mixing is computed by the first-order turbulent closure model, K-Profile Parameterization (KPP).

Uchiyama, Y., McWilliams, J.C. and Shchepetkin, A.F. (2010): Wave-current interaction in an oceanic circulation model with a vortex force formalism: Application to the surf zone. *Ocean Modelling* Vol. 34:1-2, pp.16-35.

```
# define SHOREFACE
```

CPP options:

```
# undef OPENMP
# undef MPI
# define SOLVE3D
# define UV_ADV
# undef MASKING
# define WET_DRY
# define NEW_S_COORD
# define ANA_GRID
# define ANA_INITIAL
# define ANA_SMFLUX
# define ANA_STFLUX
# define ANA_SSFLUX
# define ANA_SRFLUX
# define ANA_SST
# define ANA_BTFLUX
# define NS_PERIODIC
# define OBC_WEST
# define SPONGE
```

(continues on next page)

(continued from previous page)

```
# define MRL_WCI
# ifdef MRL_WCI
# undef WAVE_OFFLINE
# ifndef WAVE_OFFLINE
# define WKB_WWAVE
# define WKB_OBC_WEST
# define WAVE_FRICTION
# undef WAVE_ROLLER
# undef MRL_CEW
# endif
# endif
# define LMD_MIXING
# define LMD_SKPP
# define LMD_BKPP
# undef BBL
# undef SEDIMENT
# ifdef SEDIMENT
# define TCLIMATOLOGY
# define TNUDGING
# define ANA_TCLIMA
# endif
```

Settings :

Results :

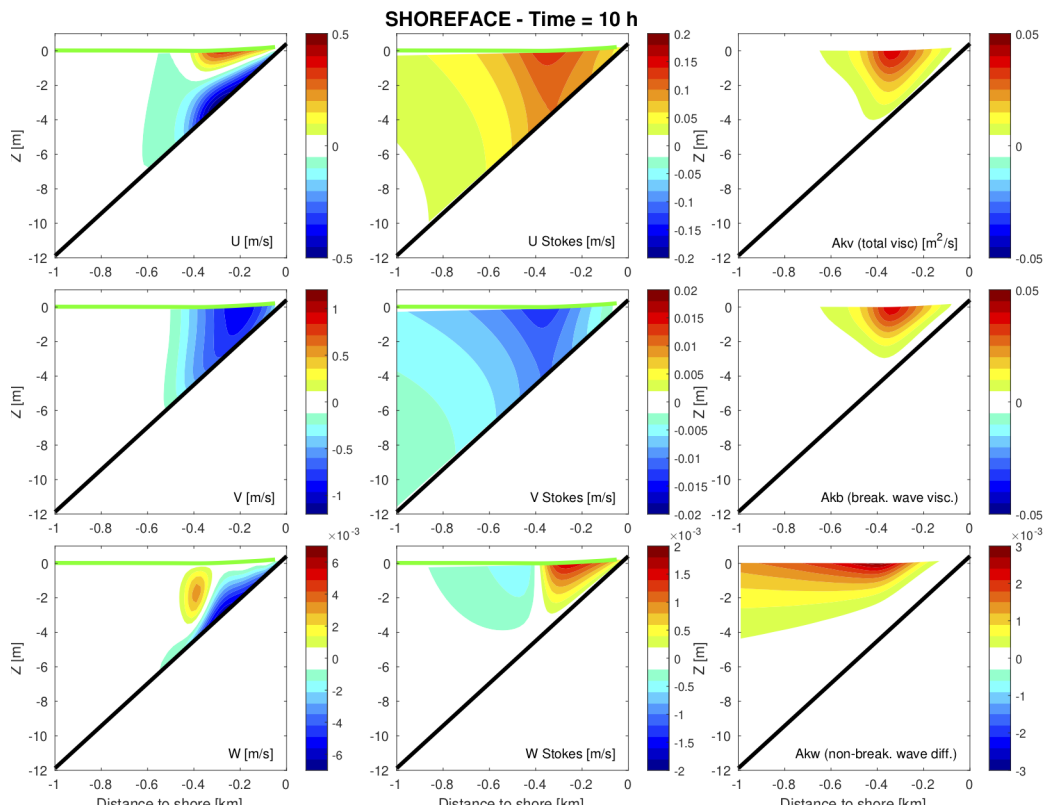


Fig. 19: SHOREFACE results : Eulerian velocities (left), Stokes velocities (center), Vertical mixing (right)

18.18 Rip Current

Rip currents are strong, seaward flows formed by longshore variation of the wave-induced momentum flux. They are responsible for the recirculation of water accumulated on a beach by a weaker and broader shoreward flow. Here, we consider longshore variation of the wave-induced momentum flux due to breaking at barred bottom topography with an imposed longshore perturbation, as in Weir et al. (2010) but in the 3D case. The basin is rectangular (768 m by 768 m) and the topography is constant over time and based on field surveys at Duck, North Carolina. Shore-normal, monochromatic waves (1m, 10s) are imposed at the offshore boundary and propagate through the WKB wave model coupled with the 3D circulation model (Uchiyama et al., 2011). The domain is periodic in the alongshore direction. We assume that the nearshore boundary is reflectionless, and there is no net outflow at the offshore boundary.

Weir, B., Y. Uchiyama, E. M. Lane, J. M. Restrepo, and J. C. McWilliams (2011), A vortex force analysis of the interaction of rip currents and surface gravity waves, *J. Geophys. Res.*, 116, C05001.

Related CPP options:

RIP	Idealized Duck Beach with 3D topography (default)
BISCA	Semi-realistic Biscarosse Beach (needs input files)
RIP_TOPO_2D	Idealized Duck with longshore uniform topography
GRANDPOPO	Idealized longshore uniform terraced beach (Grand Popo, Benin)
ANA_TIDES	Adds idealized tidal variations
WAVE_MAKER & NBQ	Wave resolving rather than wave-averaged case (#undef MRL_WCI)

CPP options:

```
# define RIP
```

```
# undef OPENMP
# undef MPI
# define SOLVE3D
# define NEW_S_COORD
# define UV_ADV
# define BSTRESS_FAST
# undef NBQ
# ifdef NBQ
# define NBQ_PRECISE
# define WAVE_MAKER
# define WAVE_MAKER_SPECTRUM
# define WAVE_MAKER_DSPREAD
# define UV_HADV_WENO5
# define UV_VADV_WENO5
# define W_HADV_WENO5
# define W_VADV_WENO5
# define GLS_MIXING_3D
# undef ANA_TIDES
# undef MRL_WCI
# define OBC_SPECIFIED_WEST
# define FRC_BRY
# define ANA_BRY
# define Z_FRC_BRY
# define M2_FRC_BRY
# define M3_FRC_BRY
# define T_FRC_BRY
# define AVERAGES
# define AVERAGES_K
# else
# define UV_VIS2
# define UV_VIS_SMAGO
# define LMD_MIXING
```

(continues on next page)

(continued from previous page)

```

# define LMD_SKPP
# define LMD_BKPP
# define MRL_WCI
# endif
# define WET_DRY
# ifdef MRL_WCI
# define WKB_WWAVE
# define WKB_OBC_WEST
# define WAVE_ROLLER
# define WAVE_FRICTION
# define WAVE_STREAMING
# define MRL_CEW
# ifdef RIP_TOPO_2D
# define WAVE_RAMP
# endif
# endif
# ifndef BISCA
# define ANA_GRID
# endif
# define ANA_INITIAL
# define ANA_SMFLUX
# define ANA_STFLUX
# define ANA_SSFLUX
# define ANA_SRFLUX
# define ANA_SST
# define ANA_BTFLUX
# if !defined BISCA && !defined ANA_TIDES
# define NS_PERIODIC
# else
# define OBC_NORTH
# define OBC_SOUTH
# endif
# define OBC_WEST
# define SPONGE
# ifdef ANA_TIDES
# define ANA_SSH
# define ANA_M2CLIMA
# define ANA_M3CLIMA
# define ZCLIMATOLOGY
# define M2CLIMATOLOGY
# define M3CLIMATOLOGY
# define M2NUDGING
# define M3NUDGING
# endif
# ifdef BISCA
# define BBL
# endif
# undef SEDIMENT
# ifdef SEDIMENT
# define ANA_SEDIMENT
# undef ANA_SPFLUX
# undef ANA_BPFLUX
# endif
# undef DIAGNOSTICS_UV

```

Setting :

Results :

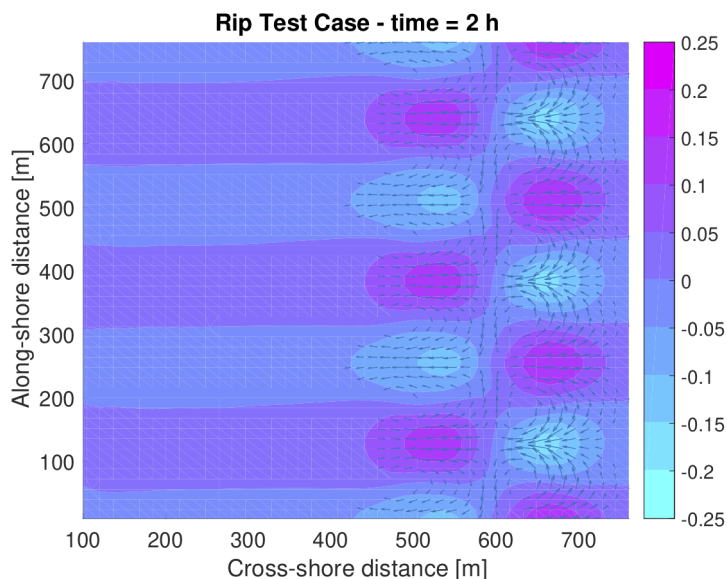


Fig. 20: RIP results : velocity

18.19 Sandbar

In this test case, we attempt to reproduce the results of a wave flume experiment conducted in a wave flume at Delft Hydraulics and reported in Roelvink and Stive (1989). The main dimensions of the flume are a length of 55 m, width of 1 m, and height of 1 m. The experiments were designed to reveal the process of bar formation and seaward bar progression. The beach configuration consists of an initially plane beach (1/40 slope). In the experiment, waves of Jonswap type are transmitted to the beach profile, creating a surf zone characteristic of a dissipative beach. The offshore RMS wave height (twice the amplitude) is 12.3 cm and peak period is 2s. The WKB model reproduces the shoreward propagation of these waves with breaking dissipation formulation by Church and Thornton (1993), using $\gamma = 0.3$ and $Br = 0.6$. Note that Stokes drift is neglected here as it is underestimated in wave flumes of this dimension due to interactions with the wave-maker and lateral walls (Huang et al., 2007).

The model sediment parameters are derived from semi-empirical relationships. The beach consists of medium to fine quartz sediment of 100 μm median grain diameter. Sinking velocity is estimated at 6 mm/s using the relation of Soulsby (1995). The erosion parameter is $1.10^{-4} kg.m^{-2}.s^{-1}$ (in the lower limit of the relation by Smith and McLean, 1977), and the shear stress is relatively low at $0.003 N.m^{-2}$. We also make use here of the morphological acceleration factor (Morfac) for speeding up sediment dynamics and saving CPU time (Roelvink, 2006). We use a factor 24, so that 1 hour of simulation translates into a full day of bed evolution (the test shows insignificant differences between simulations with and without acceleration). For validation, we compare at 12h and 24h the model results with measurements of wave height, undertow (horizontal flow 5 cm above the bottom; error ~ 1 cm/s), and bed evolution.

Roelvink, J.A. and Stive, M.J.F., 1989: Bar-generating cross-shore flow mechanisms on a beach. Journal of Geophysical Research

```
# define SANDBAR
```

CPP options:

```
# undef OPENMP
# undef MPI
# define SOLVE3D
# define UV_ADV
# define NEW_S_COORD
# define ANA_GRID
```

(continues on next page)

(continued from previous page)

```
# define ANA_INITIAL
# define ANA_SMFLUX
# define ANA_STFLUX
# define ANA_SFLUX
# define ANA_SRFLUX
# define ANA_SST
# define ANA_BTFLUX
# define NS_PERIODIC
# define OBC_WEST
# define SPONGE
# define WET_DRY
# define MRL_WCI
# ifdef MRL_WCI
#   define WKB_WWAVE
#   define MRL_CEW
#   define WKB_OBC_WEST
#   define WAVE_ROLLER
#   define WAVE_FRICTION
#   define WAVE_BREAK_SWASH
#   undef WAVE_STREAMING
#   undef WAVE_RAMP
# endif
# define LMD_MIXING
# define LMD_SKPP
# define LMD_BKPP
# define LMD_VMIX_SWASH
# undef GLS_MIXING
# define BBL
# define SEDIMENT
# ifdef SEDIMENT
#   define TCLIMATOLOGY
#   define TNUDGING
#   define ANA_TCLIMA
#   define MORPHODYN
# endif
# define NO_FRCFILE
```

Setting :

Results :

18.20 Swash

Wave dynamics on a gently sloping laboratory beach (Globex experiment), using a wave-resolving configuration.

Michallet H., B. G. Ruessink, M.V.L.M. da Rocha, A. de Bakker, D. van der A, et al.. GLOBEX: Wave dynamics on a shallow sloping beach. HYDRALAB IV Joint User Meeting, Lisbon, July 2014, Jul 2014, Lisbonne, Portugal.

```
# define SWASH
```

CPP options:

```
# define SWASH_GLOBEX_B2
# undef SWASH_GLOBEX_A3
# undef OPENMP
# undef MPI
# define SOLVE3D
# define AVERAGES
# define NBQ
# define NBQ_PRECISE
```

(continues on next page)

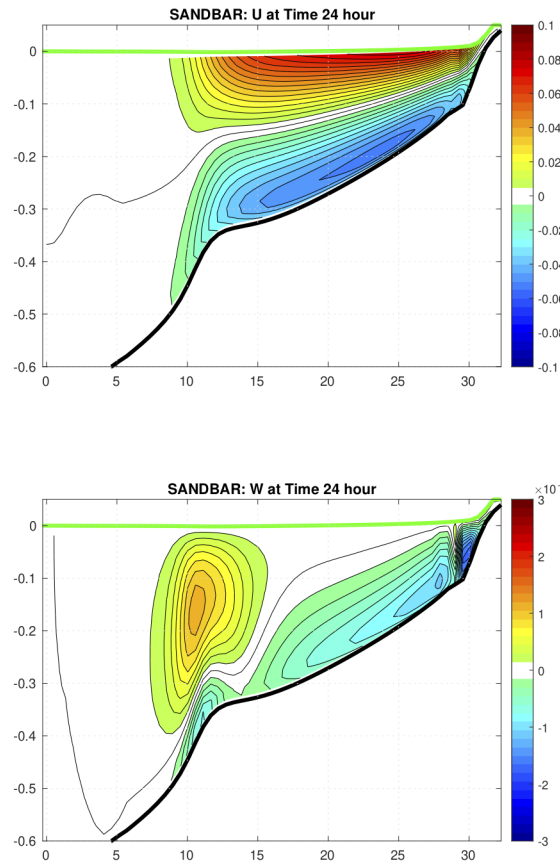


Fig. 21: SANDBAR results : undertow

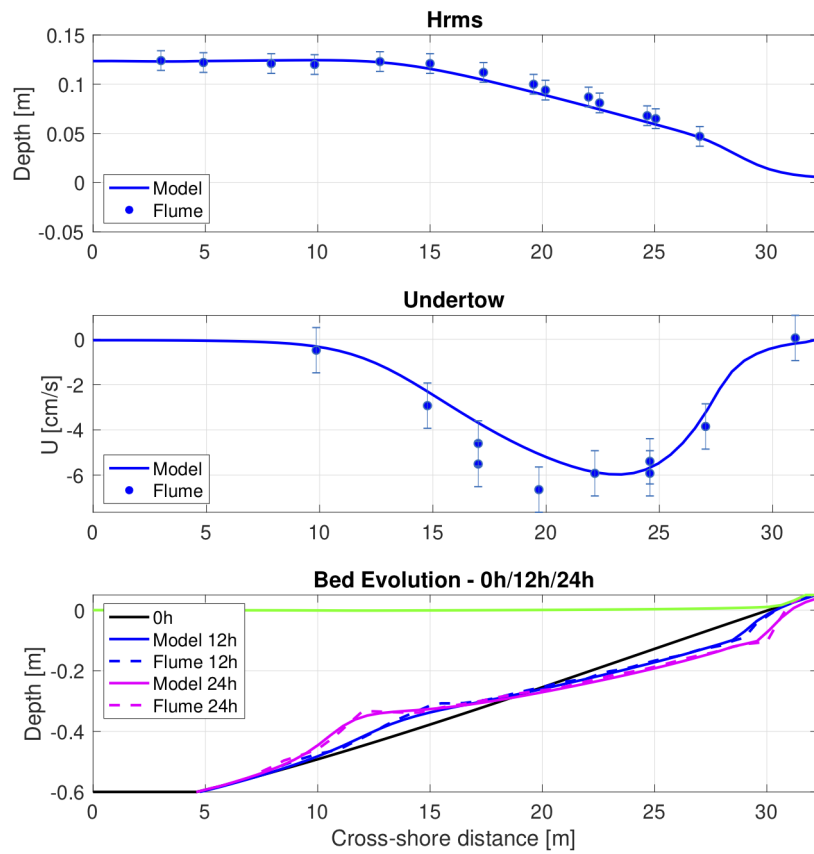


Fig. 22: SANDBAR results : validation against flume experiment

(continued from previous page)

```
# define WAVE_MAKER
# define UV_ADV
# define UV_HADV_WENO5
# define UV_VADV_WENO5
# define W_HADV_WENO5
# define W_VADV_WENO5
# define GLS_MIXING_3D
# define NEW_S_COORD
# define ANA_GRID
# define ANA_INITIAL
# define ANA_SMFLUX
# define ANA_STFLUX
# define ANA_SSFLUX
# define ANA_SRFLUX
# define ANA_SST
# define ANA_BTFLUX
# define OBC_WEST
# define OBC_SPECIFIED_WEST
# define FRC_BRY
# define ANA_BRY
# define Z_FRC_BRY
# define M2_FRC_BRY
# define M3_FRC_BRY
# define T_FRC_BRY
# define WET_DRY
# define NO_FRCFILE
```

Settings :

Results :

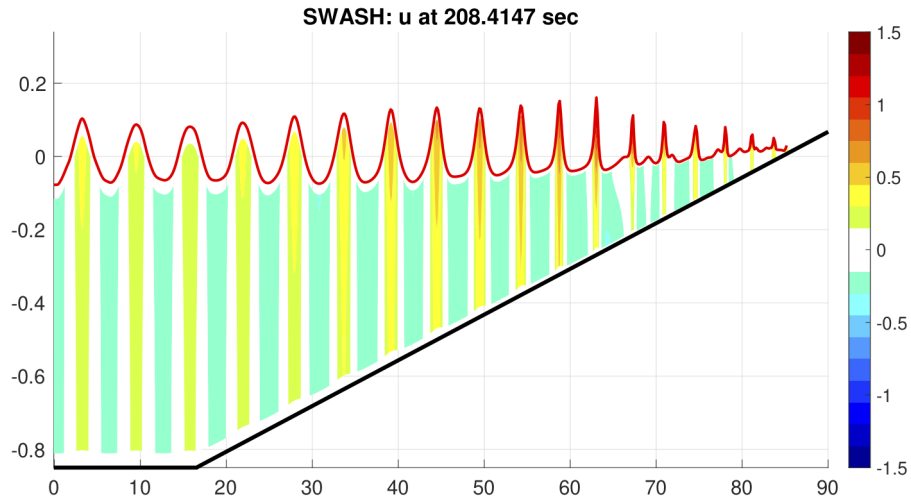


Fig. 23: SWASH results : Velocity and elevation

18.21 Tank

Chen, X.J., 2003. A fully hydrodynamic model for three-dimensional, free-surface flows. Int. J. Numer. Methods Fluids 42

```
# define TANK
```

CPP options:

```
# undef MPI
# define NBQ
# ifdef NBQ
# undef NBQ_PRECISE
# endif
# define SOLVE3D
# undef UV_ADV
# define NEW_S_COORD
# define ANA_GRID
# define ANA_INITIAL
# define ANA_BTFLUX
# define ANA_SMFLUX
# define ANA_SRFLUX
# define ANA_STFLUX
# define NO_FRCFILE
```

Settings :

Results :

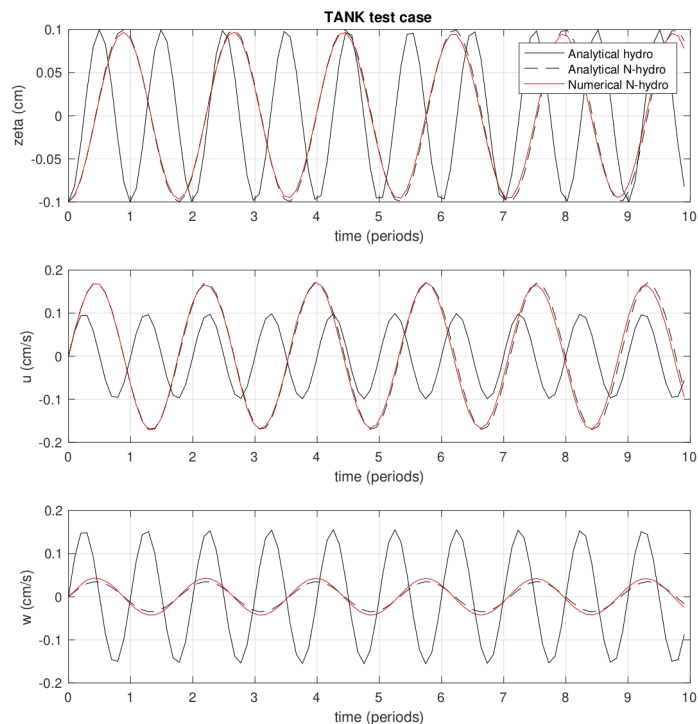


Fig. 24: TANK results : Comparison between analytical, hydrostatique and non-hydrostatique solutions

18.22 Acoustic wave

```
# define ACOUSTC
```

CPP options:

```
# undef MPI
# define NBQ
# ifdef NBQ
#   undef NBQ_PRECISE
#   define NBQ_PERF
# endif
# undef UV_VIS2
# define SOLVE3D
# define NEW_S_COORD
# define ANA_GRID
# define ANA_INITIAL
# define ANA_SMFLUX
# define ANA_STFLUX
# define ANA_SRFLUX
# define ANA_BTFLUX
# define NO_FRCFILE
```

18.23 Gravitational Adjustment

```
# define GRAV_ADJ
```

CPP options:

```
# undef OPENMP
# undef MPI
# undef NBQ
# undef XIOS
# define UV_VIS2
# define SOLVE3D
# define NEW_S_COORD
# define UV_ADV
# define TS_HADV_WENO5
# define TS_VADV_WENO5
# define ANA_GRID
# define ANA_INITIAL
# define ANA_SMFLUX
# define ANA_STFLUX
# define ANA_BTFLUX
# undef PASSIVE_TRACER
# define NO_FRCFILE
```

Settings :

Results :

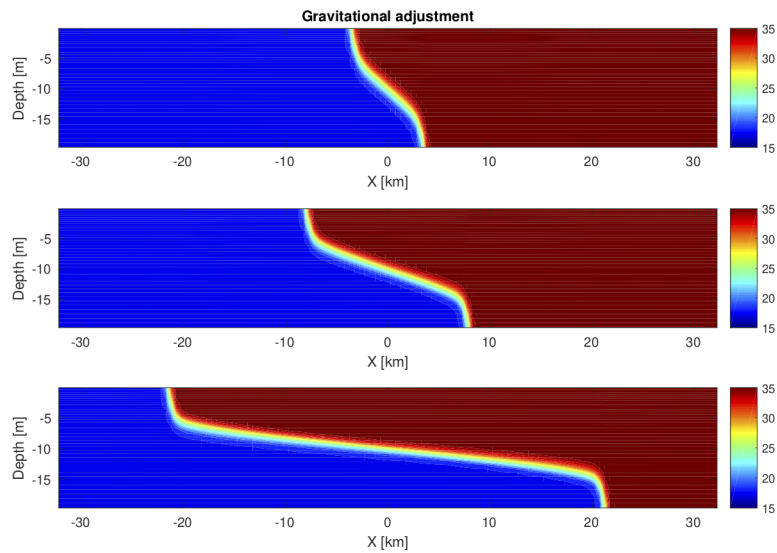


Fig. 25: GRAV_ADJ results : density front evolution

18.24 Internal Soliton

Internal soliton case (non-hydro test) is setup from: Horn, D.A., J. Imberger, & G.N. Ivey, (2001). The degeneration of large-scale interfacial gravity waves in lakes. *J. Fluid Mech.*, 434:181-207.

```
# define I_SOLITON
```

CPP options:

```
# undef OPENMP
# undef MPI
# define NBQ
# undef XIOS
# define UV_VIS2
# define SOLVE3D
# define NEW_S_COORD
# define UV_ADV
# define TS_HADV_WENO5
# define TS_VADV_WENO5
# define ANA_GRID
# define ANA_INITIAL
# define ANA_SMFLUX
# define ANA_STFLUX
# define ANA_BTFLUX
# undef PASSIVE_TRACER
# define NO_FRCFILE
```

Settings :

Results :

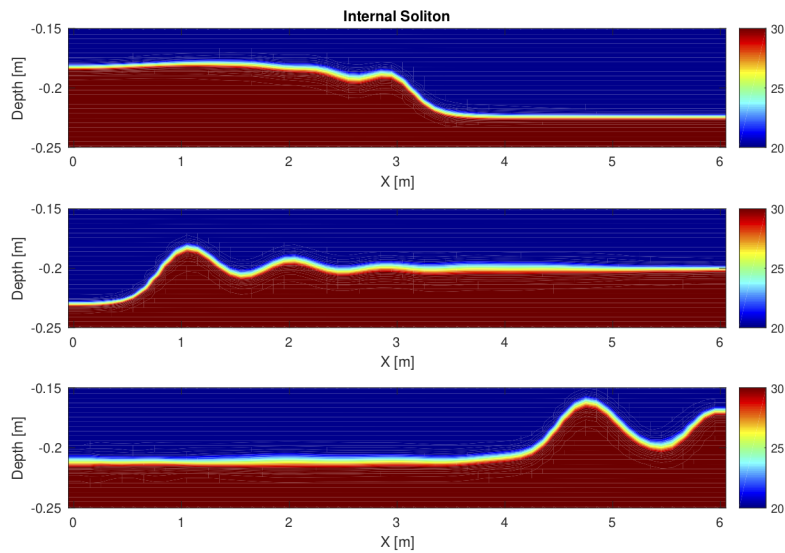


Fig. 26: I_SOLITON results : internal wave generation

18.25 Kevlin-Helmoltz Instability

```
# define KH_INST
```

CPP options:

```
# undef KH_INSTY
# undef KH_INST3D
# define MPI
# define NBQ
# define NBQ_PRECISE
# undef XIOS
# define SOLVE3D
# define NEW_S_COORD
# define UV_ADV
# define TS_HADV_WENO5
# define TS_VADV_WENO5
# define UV_HADV_WENO5
# define UV_VADV_WENO5
# define W_HADV_WENO5
# define W_VADV_WENO5
# undef SALINITY
# undef PASSIVE_TRACER
# define ANA_GRID
# define ANA_INITIAL
# define ANA_SMFLUX
# define ANA_STFLUX
# undef ANA_SRFLUX
# define ANA_BTFLUX
# define ANA_SSFLUX
# define ANA_BSFLUX
# ifndef KH_INSTY
# define EW_PERIODIC
# else
# define NS_PERIODIC
# endif
# define NO_FRCFILE
```

Setting :

Results :

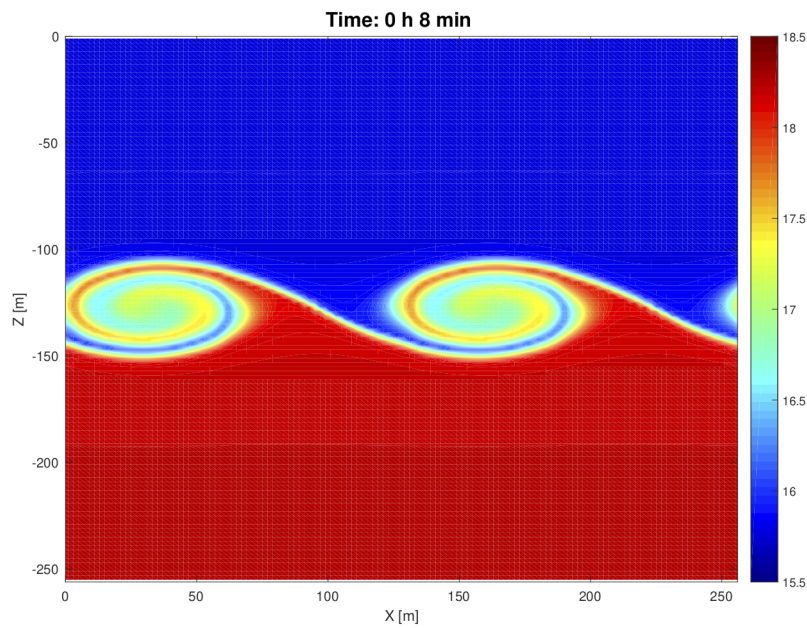


Fig. 27: KH_INST results : instability generation

18.26 Horizontal tracer advection

Test CROCO horizontal advection schemes for tracers

SOLID_BODY_ROT Example with spatially varying velocity
 DIAGONAL_ADV Constant advection in the diagonal
 SOLID_BODY_PER Example with a space and time-varying velocity

```
# define TS_HADV_TEST
```

CPP options:

```
# undef SOLID_BODY_ROT
# undef DIAGONAL_ADV
# define SOLID_BODY_PER

# undef OPENMP
# undef MPI
# undef UV_ADV
# define NEW_S_COORD
# undef UV_COR
# define SOLVE3D
# define M2FILTER_NONE
# define ANA_VMIX
# define ANA_GRID
# define ANA_INITIAL
# define ANA_SMFLUX
# define ANA_SRFLUX
# define ANA_STFLUX
# define ANA_BTFLUX
# define ANA_BSFLUX
# define ANA_SSFLUX
# define NO_FRCFILE
# define SALINITY
# define EW_PERIODIC
```

(continues on next page)

(continued from previous page)

```
# define NS_PERIODIC
# define TS_HADV_UP3
# undef TS_HADV_C4
# undef TS_HADV_UP5
# undef TS_HADV_WENO5
# undef TS_HADV_C6
```


APPENDICES

19.1 cppdefs.h

This file defines the CPP keys that are used by the the C-preprocessor when compiling CROCO. The C-preprocessor selects the different parts of the Fortran code which needs to be compiled depending on the defined CPP options. These options are separated in two parts: the basic option keys in `cppdefs.h` and the advanced options keys in `cppdefs_dev.h`.

CPP keys define the case: test case, realistic case, as well as the numerical schemes, parameterizations, and modules used, and the forcing and boundary conditions.

- **Configuration**

CPP options	Description
BASIN	Must be defined for running the Basin example
CANYON_A	Must be defined for running the Canyon_A example
CANYON_B	Must be defined for running the Canyon_B example
EQUATOR	Must be defined for running the Equator example
GRAV_ADJ	Must be defined for running the Gravitational Adjustment example
ACOUSTIC	Must be defined for running the acoustic example
INNERSHELF	Must be defined for running the Inner Shelf example
OVERFLOW	Must be defined for running the Gravitational/Overflow example
SEAMOUNT	Must be defined for running the Seamount example
SHELFROUNT	Must be defined for running the Shelf Front example
SOLITON	Must be defined for running the Equatorial Rossby Wave example
UPWELLING	Must be defined for running the Upwelling example
INTERNAL	Must be defined for running the Internal tides example
VORTEX	Must be defined for running the Baroclinic Vortex example
JET	Must be defined for running the Jet example
THACKER	Must be defined for running the Thacker example
TANK	Must be defined for running the Tank example
S2DV	Must be defined for running the S2DV example
RIP	Must be defined for running the Rip current example
SHOREFACE	Must be defined for running the Shoreface example
SWASH	Must be defined for running the Swash example
REGIONAL	Must be defined if running realistic regional simulations

- **Prallelisation**

CPP options	Description
OPENMP	Activate Open-MP parallelization protocol
MPI	Activate MPI parallelization protocol
PARALLEL_FILES	Activate parallel I/O writing
XIOS	Use external server for output
AUTOTILING	Activate subdomains partitionning optimization

- Nesting

CPP options	Description
AGRIF	Activate nesting capabilities (1-WAY by default)
AGRIF_2WAY	Activate 2-WAY nesting (update parent solution by child solution)

- Open Boundary Conditions

CPP options	Description
OBC_EAST	Open eastern boundary
OBC_WEST	Open western boundary
OBC_SOUTH	Open southern boundary
OBC_NORTH	Open northern boundary

- Tides

CPP options	Description
TIDES	Activate tidal forcing at open boundaries
SSH_TIDES	process and use tidal sea level data
UV_TIDES	process and use tidal current data
TIDERAMP	Apply ramping on tidal forcing (1 day) at initialization Warning! This should be undefined if restarting the model

- Applications

CPP options	Description
BIOLOGY	Activate biogeochemical modeling
FLOATS	Activate floats
STATIONS	Store high frequency model outputs at stations
PASSIVE_TRACER	Add a passive tracer
SEDIMENT	Activate sediment modeling
BBL	Activate bottom boundary layer parametrization

- Grid Configuration

CPP options	Description
CURVGRID	Activate curvilinear coordinate transformation
SPHERICAL	Activate longitude/latitude grid positioning
MASKING	Activate land masking
WET_DRY	Activate wetting-Drying scheme
NEW_S_COORD	Choose new vertical S-coordinates

- Model Dynamics

CPP options	Description
SOLVE3D	solve 3D primitive equations
UV_COR	Activate Coriolis terms
UV_ADV	Activate advection terms
NBQ	Activate non-boussinesq option

- Lateral Momentum Advection

CPP options	Description
DV_UP3	Activate 3rd-order upstream biased advection scheme
UV_HADV_UP5	Activate 5th-order upstream biased advection scheme
UV_HADV_C2	Activate 2nd-order centred advection scheme (should be used with explicit momentum mixing)
UV_HADV_C4	Activate 4th-order centred advection scheme (should be used with explicit momentum mixing)
UV_HADV_C6	Activate 6th-order centred advection scheme (should be used with explicit momentum mixing)
UV_HADV_WENO5	Activate WENO 5th-order advection scheme
UV_HADV_TVD	Activate Total Variation Diminishing scheme

- **Lateral Momentum Mixing**

CPP options	Description
UV_MIX_GEO	Activate mixing on geopotential (constant depth) surfaces
UV_MIX_S	Activate mixing on iso-sigma (constant sigma) surfaces
UV_VIS2	Activate Laplacian horizontal mixing of momentum
UV_VIS4	Activate Bilaplacian horizontal mixing of momentum
UV_VIS_SMAGO	Activate Smagorinsky parametrization of turbulent viscosity (only with UV_VIS2)
UV_VIS_SMAGO3D	Activate 3D Smagorinsky parametrization of turbulent viscosity

- **Lateral Tracer Advection**

CPP options	Description
TS_HADV_UP3	3rd-order upstream biased advection scheme
TS_HADV_RSUP3	Split and rotated 3rd-order upstream biased advection scheme
TS_HADV_UP5	5th-order upstream biased advection scheme
TS_HADV_RSUP5	Split and rotated 5th-order upstream biased advection scheme with reduced dispersion/diffusion
TS_HADV_C4	4th-order centred advection scheme
TS_HADV_C6	Activate 6th-order centred advection scheme
TS_HADV_WENO5	5th-order WENOZ quasi-monotonic advection scheme for all tracers
BIO_HADV_WENO5	5th-order WENOZ quasi-monotone advection scheme for passive tracers (including biology and sediment tracers)

• **Lateral Tracer Mixing**

CPP options	Description
TS_MIX_ISO	Activate mixing along isopycnal (isoneutral) surfaces
TS_MIX_GEO	Activate mixing along geopotential surfaces
TS_MIX_S	Activate mixing along iso-sigma surfaces
TS_DIF2	Activate Laplacian horizontal mixing of tracer
TS_DIF4	Activate Bilaplacian horizontal mixing of tracer
TS_MIX_IMP	Activate stabilizing correction of rotated diffusion (used with TS_MIX_ISO and TS_MIX_GEO)

• **Nudging**

CPP options	Description
ZNUDGING	Activate nudging layer for zeta.
M2NUDGING	Activate nudging layer for barotropic velocities.
M3NUDGING	Activate nudging layer for baroclinic velocities.
TNUDGING	Activate nudging layer for tracer.
ROBUST_DIAG	Activate strong tracer nudging in the interior for diagnostic simulations

• **Vertical Mixing**

CPP options	Description
BODYFORCE	Apply surface and bottom stresses as body-forces
ANA_VMIX	Activate analytical viscosity/diffusivity coefficients
BVF_MIXING	Activate a simple mixing scheme based on the Brunt-Väisälä frequency
LMD_MIXING	Activate Large/McWilliams/Doney mixing (turbulent closure for interior and planetary boundary layers) with following options

LMD_SKPP	Activate surface boundary layer KPP mixing
LMD_BKPP	Activate bottom boundary layer KPP mixing
LMD_RIMIX	Activate shear instability interior mixing
LMD_CONVEC	Activate convection interior mixing
LMD_DDMIX	Activate double diffusion interior mixing
LMD_NONLOCAL	Activate nonlocal transport for SKPP

GLS_MIXING	Activate Generic Length Scale scheme as implemented by Warner et al. (2005), default is k-kl (see below)
GLS_MIX2017	Activate Generic Length Scale scheme with a slightly different implementation (under test), default is k-epsilon (see below)
GLS_KKL	Activate K-KL (K=TKE; L=length Scale) as in Mellor-Yamada 2.5 (1974)
GLS_KOMEGA	Activate K-OMEGA (OMEGA=frequency of TKE dissipation) originating from Kolmogorov (1942)
GLS_KEPSILON	Activate K-EPSILON (EPSILON=TKE dissipation) as in Jones and Launder (1972)
GLS_GEN	Activate generic model of Umlauf and Burchard (2003)

- **Equation of State**

CPP options	Description
SALINITY	Activate salinity as an active tracer
NONLIN_EOS	Activate nonlinear equation of state
SPLIT_EOS	Activate the split of the nonlinear equation of state in adiabatic and compressible parts for reduction of pressure gradient errors

• **Surface Forcing**

CPP options	Description
BULK_FLUX	Activate bulk formulation for surface heat fluxes
BULK_FAIRALL	Choose Fairall formulation (default: COAMPS formulation)
BULK_EP	Add in bulk formulation for fresh water fluxes
BULK_LW	Add in long-wave radiation feedback from model SST
BULK_SMFLUX	Add in bulk formulation for surface momentum fluxes
SST_SKIN	Activate skin sst computation (Zeng & Beljaars, 2005)
ONLINE	Read native files and perform online interpolation on ROMS grid (default cubic interpolation)
QCORRECTION	Activate linearized bulk formulation providing heat flux correction dQdSST for nudging towards model SST
SFLX_CORR	Activate freshwater flux correction around model SSS
ANA_DIURNAL_SW	Activate analytical diurnal modulation of short wave radiations (only appropriate if there is no diurnal cycle in data)

• **Coupling**

CPP options	Description
OW_COUPLING	Activate Ocean-Wave coupling
OA_COUPLING	Activate Ocean-Atmosphere coupling
OA_MCT	Use OASIS-MCT for coupling

• **Sponge Layer**

CPP options	Description
SPONGE	Activate areas of enhanced viscosity and diffusivity near lateral open boundaries.
SPONGE_GRID	Automatic setting of the sponge width and value
SPONGE_DIF2	Sponge on tracers (default)
SPONGE_VIS2	Sponge on momentum (default)
SPONGE_SED	Sponge on sediment (default)

• **Lateral forcing**

CPP options	Description
CLIMATOLOGY	Activate processing of 2D/3D climatological data for nudging layers and open boundary forcing
ZCLIMATOLOGY	Activate processing of sea level
M2CLIMATOLOGY	Activate processing of barotropic velocities
M3CLIMATOLOGY	Activate processing of baroclinic velocities
TCLIMATOLOGY	Activate processing of tracers
ZNUDGING	Activate nudging layer for sea level
M2NUDGING	Activate nudging layer for barotropic velocities
M3NUDGING	Activate nudging layer for baroclinic velocities
TNUDGING	Activate nudging layer for tracers
ROBUST_DIAG	Activate nudging over the whole domain
FRC_BRY	Activate processing of 1D/2D climatological or simulation/reanalysis data at open boundary points
Z_FRC_BRY	Activate open boundary forcing for sea level
M2_FRC_BRY	Activate open boundary forcing for barotropic velocities
M3_FRC_BRY	Activate open boundary forcing for baroclinic velocities
T_FRC_BRY	Activate open boundary forcing for tracers

• **Bottom Forcing**

CPP options	Description
ANA_BSFLUX	Activate analytical bottom salinity flux (generally 0)
ANA_BTFLUX	Activate analytical bottom temperature flux (generally 0)

• **Point Sources - Rivers**

CPP options	Description
PSOURCE	Activate point sources (rivers)
ANA_PSOURCE	use analytical vertical profiles for point sources (set in set_global_definitions.h)
PSOURCE_NCFILE	Read variable river transports in netcdf file
PSOURCE_NCFILE_TS	Read variable river concentration in netcdf file

• **Open boundary conditions II**

CPP options	Description
OBC_VOLCONS	Activate mass conservation enforcement at open boundaries (with OBC_M2ORLANSKI)
OBC_M2SPECIFIED	Activate specified open boundary conditions for barotropic velocities
OBC_M2ORLANSKI	Activate radiative open boundary conditions for barotropic velocities
OBC_M2FLATHER	Activate Flather open boundary conditions for barotropic velocities
OBC_M2CHARACT	Activate open boundary conditions based on characteristic methods barotropic velocities
OBC_M3SPECIFIED	Activate specified open boundary conditions for baroclinic velocities
OBC_M3ORLANSKI	Activate radiative open boundary conditions for baroclinic velocities
OBC_TSPECIFIED	Activate specified open boundary conditions for tracers
OBC_TORLANSKI	Activate radiative open boundary conditions for tracers
OBC_TUPWIND	Activate upwind open boundary conditions for tracers

• **I/O - Diagnostics**

CPP options	Description
AVERAGES	Process and output time-averaged data
AVERAGES_K	Process and output time-averaged vertical mixing
DIAGNOSTICS_TS	Store and output budget terms of the tracer equations
DIAGNOSTICS_TS_ADV	Choose advection rather than transport formulation for tracer budgets
DIAGNOSTICS_TS_MLD	Integrate tracer budgets over the mixed-layer depth
DIAGNOSTICS_UV	Store and output budget terms of the momentum equations
XIOS	Use XIOS IO server

• **Biogeochemical models**

CPP options	Description
PISCES	Activate 24-component PISCES biogeochemical model
BIO_NChlPZD	Activate 5-component NPZD type model
BIO_N2PZD2	Activate 7-component NPZD type model
BIO_BioEBUS	Activate 12-component NPZD type model

• **Sediment Dynamics**

CPP options	Description
ANA_SEDIMENT	Set analytical sediment ripple and bed parameters
ANA_WWAVE	Analytical (constant) wave (hs, Tp, Dir) values.
SUSPLOAD	Activate suspended load transport
BEDLOAD	Activate bedload transport
MORPHODYN	Activate morphodynamics
ANA_BPFLUX	Set kinematic bottom flux of sediment tracer (if different from 0)
SLOPE_NEMETH	Nemeth formulation for avalanching (Nemeth et al, 2006)
SLOPE_LESSER	Lesser formulation for avalanching (Lesser et al, 2004)
BED-LOAD_SOULSBY	Soulsby formulation for bedload (Soulsby, R.L. and J.S. Damgaard, 2005)
BEDLOAD_MPM	Meyer-Peter-Muller formulation for bedload (Meyer-Peter, E. and R. Muller, 1948)

• **Bottom Boundary Layer**

CPP options	Description
ANA_WWAVE	Set analytical wave forcing
ANA_BSEDIM	Set analytical bed parameters (if SEDIMENT is undefined)
Z0_BL	Compute bedload roughness for ripple predictor and sediment purposes
Z0_RIP	Determine bedform roughness ripple height and ripple length for sandy beds
Z0_BIO	Determine (biogenic) bedform roughness ripple height and ripple length for silty beds

• **Wave-Current interactions**

CPP options	Description
MRL_WCI	Set wave-current interaction model
ANA_WWAVE	Analytical values for waves
WAVE_OFFLINE	Set-wave offline (read from file) forcing
WKB_WWAVE	Set WKB wave model
OW_COUPLING	Set wave coupling for WWIII (to install separately)
MRL_CEW	Set current feedback on waves
WKB_OBC_WEST	Set East/West/North/South offshore forcing for WKB model
ANA_BRY_WKB	Analytical boundary wave forcing (from croco.in)
WAVE_BREAK_CT93	Thornton and Guza (1983, JGR) wave breaking (for WKB)
WAVE_BREAK_TG86	Church and Thornton (1993, Coastal Eng.) wave breaking (for WKB)
WAVE_BREAK_TG86A	Another Church and Thornton formulation
WAVE_ROLLER	Set wave roller model
WAVE_STREAMING	Set bottom wave streaming
WAVE_FRICTION	Set bottom friction in WKB model (used for streaming)
WAVE_RAMP	Set wave forcing ramp using wave_ramp parameter

19.2 croco.in

KEYWORD	DESCRIPTION
title	Configuration name
time_stepping	<p>NTIMES : Number of time-steps required for the simulation</p> <p>dt : Baroclinic time step [in s]</p> <p>NDTFAST : Number of barotropic time-steps between each baroclinic time step.</p> <p>For 2D configurations, ndtfast should be unity</p> <p>NINFO : Number of time-steps between printing of information to standard output</p>
time_stepping_nbq	<p>NDTNBQ</p> <p>CSOUND_NBQ</p> <p>VISC2_NBQ</p>
S-coord	<p>THETA_S: S-coordinate surface control parameter</p> <p>THETA_B: S-coordinate bottom control parameter</p> <p>Hc(m): Width of surface or bottom boundary layer in which higher vertical resolution is required during stretching</p>
run_start_date	run start date (used with USE_CALENDAR)
run_end_date	run end date (used with USE_CALENDAR)
output_time_steps	<p>DT_HIS(H)</p> <p>DT_AVG(H)</p> <p>DT_RST(H)</p>
grid	grid filename
forcing	forcing filename
bulk_forcing	bulk forcing filename (used with BULK_FLUX)
climatology	climatology filename (boundary and nudging, used with CLIMATOLOGY)
boundary	boundary filename (used with FRC_BRY)
initial	<p>NRREC: Switch to indicate start or re-start from a previous solution. nrrec is the time index of the initial or re-start NetCDF file assigned for initialization.</p> <p>If nrrec is negative (say nrrec = -1), the model will start from the most recent time record. That is, the initialization record is assigned internally.</p> <p>filename: Name of file containing the initial state.</p>

Continued on next page

Table 1 – continued from previous page

KEYWORD	DESCRIPTION
restart	<p>NRST: Number of time-steps between writing of re-start fields</p> <p>NRPFRST</p> <p>0: write several records every NRST time steps</p> <p>>0: create more than one file (with sequential numbers) and write NRPRST records per file</p> <p>-1: overwrite record every NRST time steps</p> <p>filename: name of restart file</p>
history	<p>LDEFHIS: flag (T/F) for writing history files</p> <p>NWRT: Number of time-steps between writing of history fields</p> <p>NRPFHIS:</p> <p>0: write several records every NWRT time steps</p> <p>>0: create more than one file (with sequential numbers) and write NRPHIS records per file</p> <p>-1: overwrite record every NWRT time steps</p> <p>filename: Name of history file</p>
averages	<p>NTSAVG: Starting timestep for the accumulation of output time-averaged data. For instance, you might want to average over the last day of a thirty-day run.</p> <p>NAVG: Number of time-steps between writing of averaged fields</p> <p>NRPFAVG:</p> <p>0: write several records every NAVG time steps</p> <p>>0: create more than one file (with sequential numbers) and write NRPFAVG records per file</p> <p>-1: overwrite record every NAVG time steps</p> <p>filename: Name of average file</p>
primary_history_fields	Flags for writing primary variables in history NetCDF file
auxiliary_history_fields	Flags for writing auxiliary variables in history NetCDF file
primary_averages	Flags for writing primary variables in average NetCDF file
auxiliary_averages	Flags for writing auxiliary variables in average NetCDF file
rho0	Mean density used in the Boussinesq equation.
lateral_visc	<p>VISC2: Laplacian background viscosity in m²/s (with UV_VIS2 CPP option)</p> <p>VISC4: Bilaplacian background viscosity in m⁴/s (with UV_VIS4 CPP option)</p>

Continued on next page

Table 1 – continued from previous page

KEYWORD	DESCRIPTION
tracer_diff2	TNU2(1:NT): Laplacian background diffusivity in m ² /s for each tracer (with TS_DIF2 CPP option)
tracer_diff4	TNU4(1:NT): Laplacian background diffusivity in m ⁴ /s for each tracer (with TS_DIF4 CPP option)
vertical_mixing	Constant vertical viscosity coefficient in m ² /s for analytical vertical mixing scheme (with ANA_VMIX CPP option)
bottom_drag	RDRG [m/s]: Drag coefficient for linear bottom stress formulation RDRG2: Drag coefficient for constant quadratic bottom stress formulation Zob [m]: Roughness length for Von-Karman quadratic bottom stress formulation Cdb_min: Minimum value of drag coefficient for Von-Karman quadratic bottom stress formulation Cdb_max: Maximum value of drag coefficient for Von-Karman quadratic bottom stress formulation.
gamma2	Free- or partial- or no-slip wall boundary condition. 1 means free slip conditions are used.
sponge	sponge parameters are only needed if SPONGE_GRID is undefined in set_global_definitions.h; otherwise, these parameters are assigned internally. X_SPONGE [m]: width of sponge layers V_SPONGE [m ² /s]: viscosity/diffusivity values in sponge layers. These values follow a cosine profile from zero interior value to V_SPONGE at the boundary.

Continued on next page

Table 1 – continued from previous page

KEYWORD	DESCRIPTION
nudg_cof	<p>TauT_in [days]: Short nudging time scale used in radiative open boundary conditions for tracer signal propagating inward the computational domain. This coefficient is used at boundary points and imposes strong nudging towards external data</p> <p>TauT_out [days]: Long nudging time scale used in radiative open boundary conditions for tracer signal propagating outward the computational domain. This coefficient is used at boundary points and imposes mild nudging towards external data. If CLIMATOLOGY is defined, it is also used in nudging layers with gradual decrease (cosine profile) from the open boundary to the inner border of the nudging layer.</p> <p>TauM_in [days]: Same as above, but for momentum equations</p> <p>TauM_out [days]: Same as above, but for momentum equations</p>
diagnostics	<p>ldefdia: flag that activates the storage of the instantaneous tracer budget terms in a diagnostic file</p> <p>nwrtdia: Number of time-steps between writing of diagnostic fields</p> <p>nrfpdia:</p> <ul style="list-style-type: none"> 0: write several records every NWRTDIA time steps >0: create more than one file (with sequential numbers) and write NRPFDIA records per file -1: overwrite record every NWRTDIA time steps <p>filename: Name of instantaneous tracer diagnostic file</p>

Continued on next page

Table 1 – continued from previous page

KEYWORD	DESCRIPTION
diag_avg	<p>ldefdia_avg: flag that activates the storage of averaged tracer budget terms in a diagnostic file</p> <p>ntsdia_avg: Starting timestep for the accumulation of output time-averaged data. For instance, you might want to average over the last day of a thirty-day run.</p> <p>nwrtdia_avg: Number of time-steps between writing of averaged diagnostic fields</p> <p>nprfdia_avg:</p> <ul style="list-style-type: none"> 0: write several records every NWRTDIA_AVG time steps >0: create more than one file (with sequential numbers) and write NRPFDIA_AVG records per file -1: overwrite record every NWRTDIA_AVG time steps <p>filename: Name of average tracer diagnostic file</p>
diag3D_history_fields	<p>flag to select which tracer equation (temp, salt, etc ...) to store in diagnostic file. These terms are 3D.</p>
diag2D_history_fields	<p>flags to select which tracer equation integrated over the mixed layer depth (cf DIAGNOSTICS_TS_MLD) to store in diagnostic file. These terms are 2D.</p>
diag3D_average_fields	same as diag3D_history_fields but for averaged fields
diag2D_average_fields	same as diag2D_history_fields but for averaged fields
diagnosticsM	<p>Same format as diagnostics but for momentum equations</p> <p>ldefdiaM:</p> <p>nwrtdiaM:</p> <p>nprfdiaM:</p> <p>filename:</p>
diagM_avg	Same format as diag_avg but for momentum equations
diagM_history_fields	flag to select which momentum equation (u,v) to store in diagnostic file. These terms are 3D.
diagM_average_fields	same as diagM_history_fields but for averaged fields
diagnosticsM_bio	Same format as diagnostics but for biogeochemical budget terms (other than advection/diffusion)
diagbio_avg	Same format as diag_avg but for biogeochemical budget terms (other than advection/diffusion)

Continued on next page

Table 1 – continued from previous page

KEYWORD	DESCRIPTION
diagbioFlux_history_fields	Flag (T or F) to select which biogeochemical tracer flux to store in diagnostic file. These terms are 3D. (For NPZD type model)
diagbioVSink_history_fields	Flag (T or F) to select which biogeochemical tracer sinking flux equation to store in diagnostic file These terms are 3D. This is for NPZD type model(BIO_NChIPZD, BIO_N2ChIPZD2 and BIO_BioEBUS), you need to follow the biogeochemical tracers order.
diagbioGasExc_history_fields	Flag (T or F) to select which biogeochemical tracer Gas exchange flux equation to store in diagnostic file. These terms are 2D.
diagbioFlux_average_fields	Same as above but averaged
diagbioVSink_average_fields	Same as above but averaged
diagbioGasExc_average_fields	Same as above but averaged
biology	Name of file containing the Iron dust forcing used in the PISCES biogeochemical model
sediments	input file: sediment parameters input file
sediment_history_fields	Flags for storing sediment fields in history file bed_thick:Thickness of sediment bed layer (m) bed_poros: Porosity of sediment bed layer (no unit) bed_fra(sand,silt): Volume fraction of sand/silt in bed layer (no unit)
bbl_history_fieldsi	Flags for storing bbl fields in history file Abed: Bed wave excursion amplitude (m) Hripple: Bed ripple length (m) Lripple: Bed ripple length (m) Zbnot: Physical hydraulic bottom roughness (m) Zbapp: Apparent hydraulic bottom roughness (m) Bostrw: Wave-induced kinematic bottom stress (m)
floats	Lagrangian floats application. Same format as diagnostics LDEFFLT NFLT NRPFFLT inpname, hisname
floats_fields	Type of fields computed for each lagrangian floats

Continued on next page

Table 1 – continued from previous page

KEYWORD	DESCRIPTION
station_fields	Fixed station application. Same format as diagnostics LDEFSTA NSTA NRPFSTA inpname, hisname
psource	Nsrc: point source number Isrc: I point source indice Jsrc: J point source indice Dsrc: Direction of point source flow (u=0,v=1) Qbar [m3/s]: Total transport at point source Lsrc: Logical switch for type of tracer to apply Tsrc: Tracer value
psource_ncfile	Nsrc: point source number Isrc: I point source indice Jsrc: J point source indice Dsrc: Direction of point source flow (u=0,v=1) qbardir: Orientation: South=0 or North=0, East=0 or West=1 Lsrc: Logical switch for type of tracer to apply Tsrc: Tracer value in case of analytical value [#undef PSOURCE_NCFILE_TS] runoff file name: Input netCDF runoff file

19.3 Comparison of ROMS and CROCO versions

Models	CROCO	ROMS-UCLA	ROMS-Rutgers / COASWT
Origin	UCLA-IRD-INRIA-IFREMER-SHOM-CNRS	UCLA	UCLA-Rutgers-USGS
Maintenance	IRD-INRIA-IFREMER-SHOM-CNRS	UCLA	Rutgers-USGS
Realm	Europe-World	US West Coast	US East Coast
Introductory year	1999 (AGRIF) 2016 (CROCO)	2002	1998

CODE FEATURES

Models	CROCO	ROMS-UCLA	ROMS-Rutgers / COASWT
Parallelization	MPI or OpenMP (Hybrid branch exists)	Hybrid MPI-OpenMP	MPI or OpenMP
Nesting	Online at barotropic level	Off-line (On-line at baroclinic level and not yet operational)	Off-line
Data assimilation	3DVAR	3DVAR	4DVAR
Wave-current interact.	McWilliams et al. 2004	McWilliams et al. 2004	Mellor (2003) and Mc- Williams et al. (2004)
Air-sea coupling	OASIS-MCT	Home made	MCT
Sediment Dynamics	Blaas et al. (2007), MUSTANG	Blaas et al. (2007)	Blaas et al. (2007), Warner et al. (2008)
Biogeochemistry	NPZD Gruber et al. (2006), PISCES	NPZD Gruber et al. (2006)	EcoSim, NEMURO, NPZD Franks, NPZD Powell, Fennel
Sea ice	none	none	Budgell (2005)
Vertical mixing	KPP, GLS	KPP, GLS	KPP, GLS
Wetting-Drying	Warner et al. (2013)	none	Warner et al. (2013)

TIME STEPPING

Models	CROCO	ROMS-UCLA	ROMS-Rutgers / COASWT
2D momentum	Generalized FB AB3-AM4	Generalized FB AB3-AM4	LF-AM3 with FB feed-back
3D momentum	LF-AM3	LF-AM3	AB3
Tracers	LF-AM3 with stabilizing correction for isopycnal hyperdiffusion	LF-AM3 with stabilizing correction for isopycnal hyperdiffusion	LF-TR with explicit geopotential diffusion (no stabilizing correction: strong stability constraint)
Internal waves	LF-AM3 with FB feed-back	LF-AM3 with FB feed-back	Generalized FB (AB3-TR)
Coupling stage	Predictor	Corrector	

STABILITY CONSTRAINTS (Max Courant number)

Models	CROCO	ROMS-UCLA	ROMS-Rutgers / COASWT
2D	1.78	1.78	1.85
3D advection	1.58	1.58	0.72
Coriolis	1.58	1.58	0.72
Internal waves	1.85	1.85	1.14