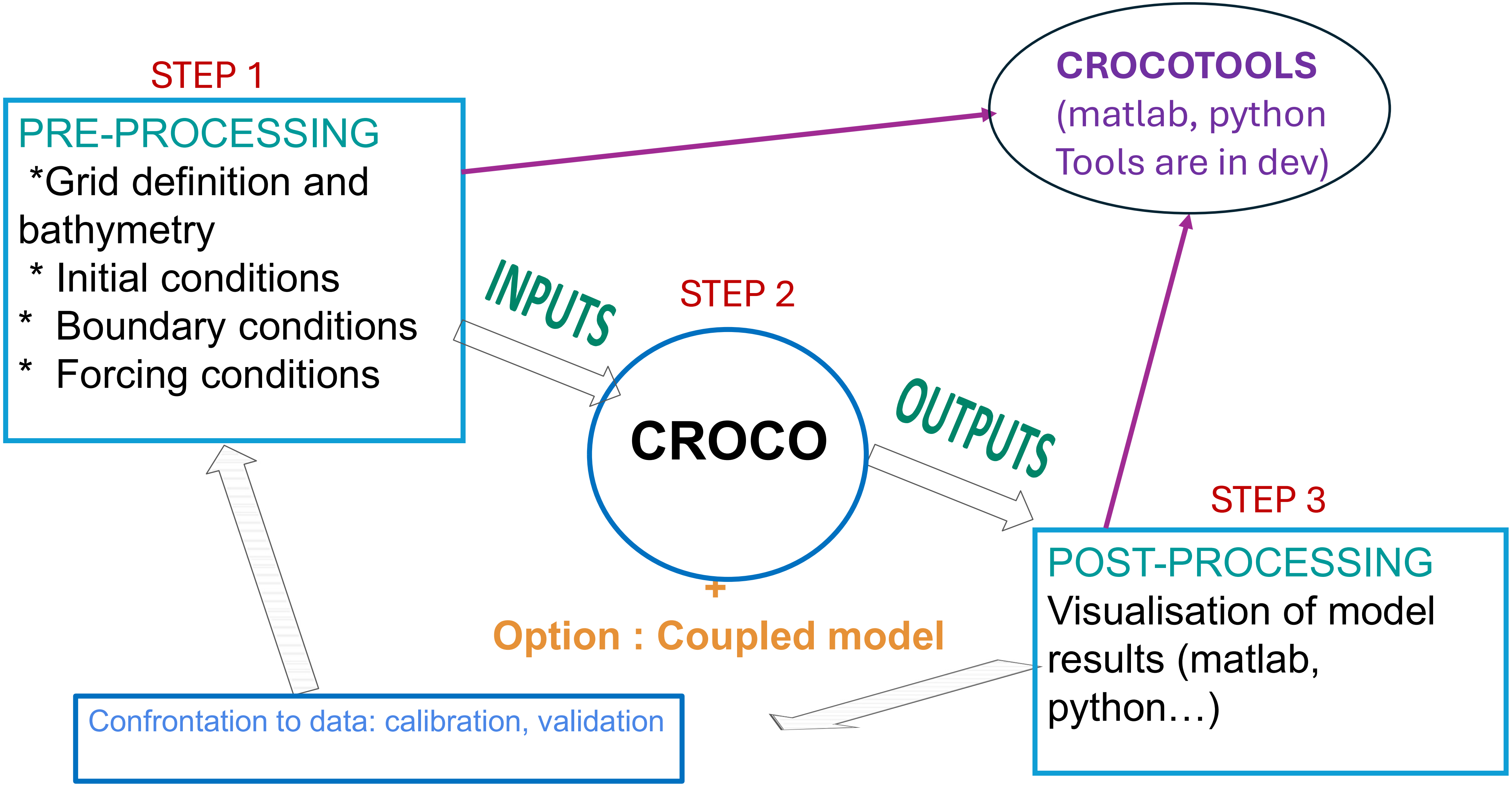
How to set a configuration on CROCO model

Coastal and Regional Ocean COmmunity model

Jihene Abdennadher & Moncef Boukthir

HOW TO PROCEED?



Croco Philosophy

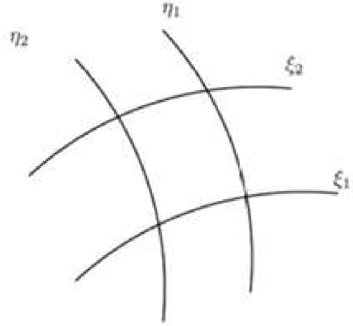
CROCO is based on a key logic: each term in the model equations corresponds to one or more keys, named **CPP options.**

CPP OPTIONS must be specified in the file cppdefs.h which is linked to the makefile

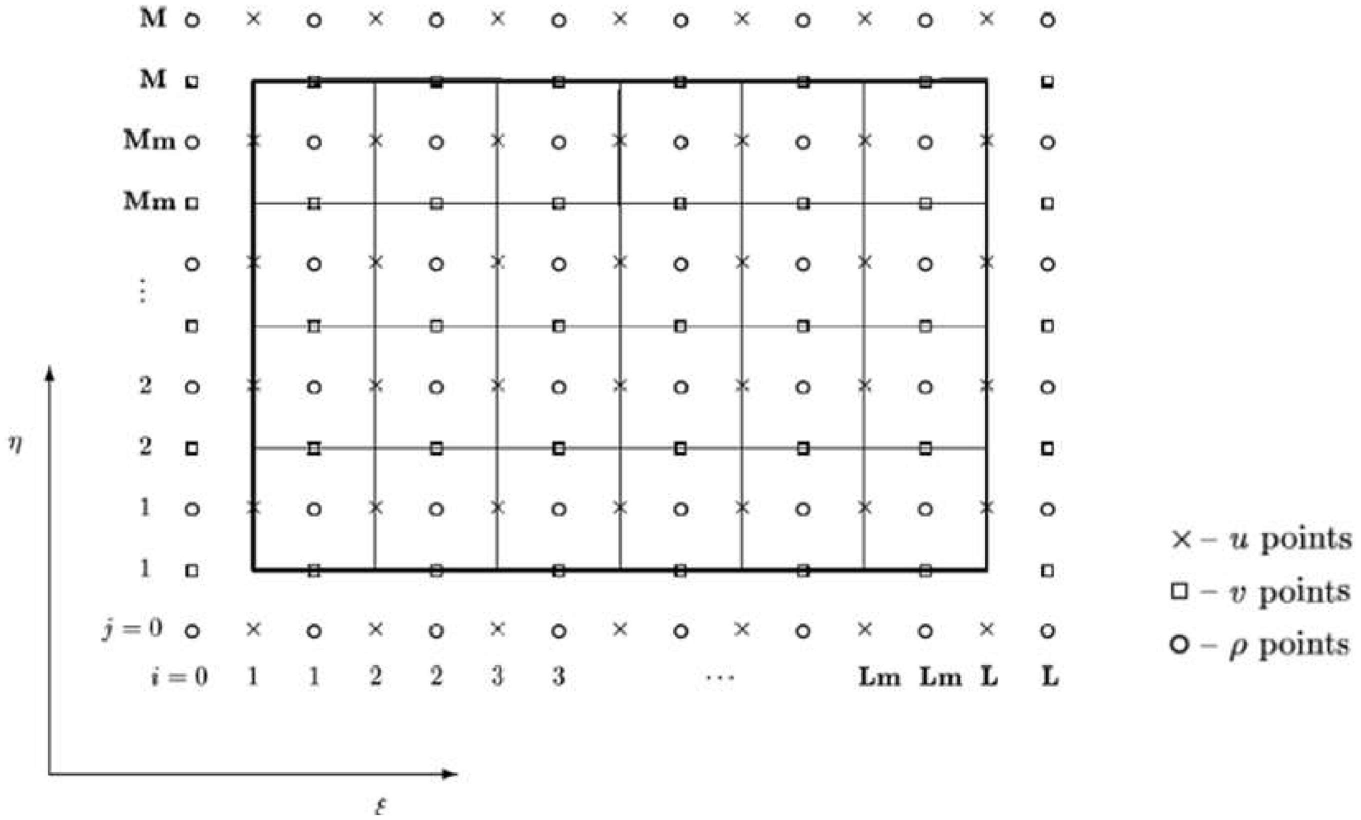
Each modification in an include file (*.h) requires recompiling the source code.

All values assigned to variables, as well as paths to grid, initialization, boundary, and forcing files, must be specified in the *croco.in* file. No source code recompilation is needed when making changes to this file

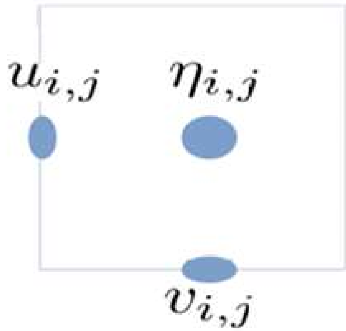
GRID



Arakawa-C structured grid



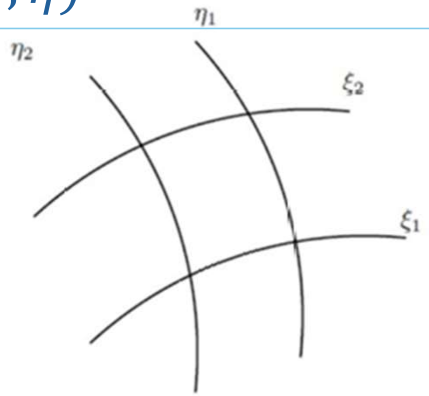
CROCO indexing



GRID

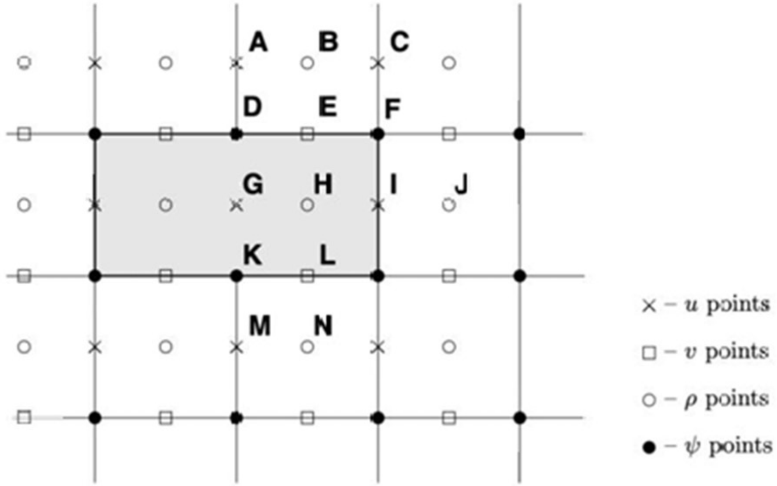
Param.h & cppdefs.h & croco.in

Orthogonal curvilinear horizontal coordinates (ξ, η)



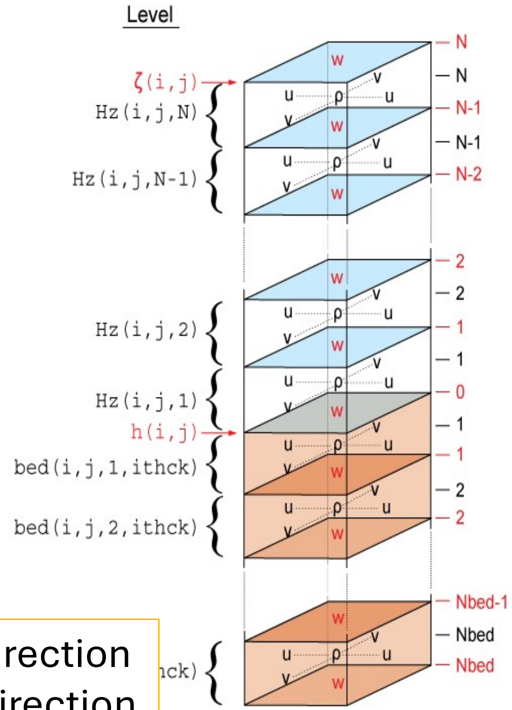
define CURVGRID
define SPHERICAL

Land/ sea masking



define MASKING

Or **#define ANA_GRID**



L: number of ρ points in ξ direction
M: number of ρ points in η direction
N: number of ρ vertical levels

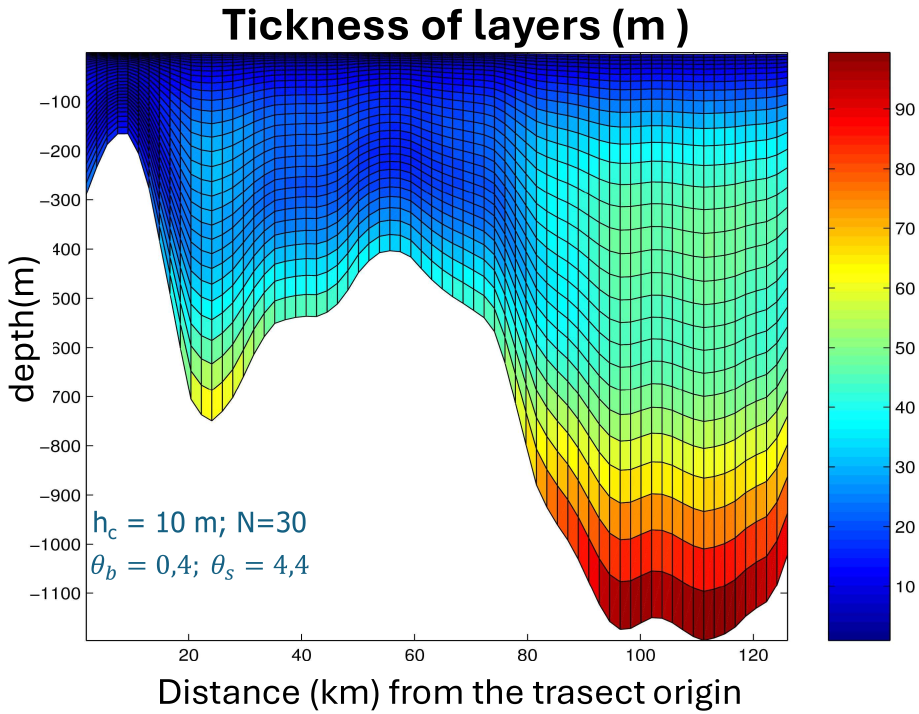
Lm (L-2) , Mm (M-2) and N **must be specified** in **param.h**
*** Path of grid file must be specified in croco.in**

param.h also contains: Parallelisation settings, Tides, Wetting-Drying, Point sources, Floats, Stations specifications

VERTCIAL GRID parameters

hc, theta_s and theta_b values must be specified in **croco.in**

Values of these parameters can be tested through the matlab script **Test_vgrid.m (see preprocessing and post-processing course)**



CROCO Equations and CPP options

cppdefs.h

Coastal and Regional Ocean COmmunity model

Momentum conservation

$$\frac{\partial u}{\partial t} + \vec{u}.\nabla u - fv = -\frac{1}{\rho_0}\frac{\partial P}{\partial x} + \nabla_h\left(K_{Mh}.\nabla_h u\right) + \frac{\partial}{\partial z}\left(K_{Mv}\frac{\partial u}{\partial z}\right)$$

$$\frac{\partial v}{\partial t} + \vec{u}.\nabla v + fu = -\frac{1}{\rho_0}\frac{\partial P}{\partial y} + \nabla_h\left(K_{Mh}.\nabla_h v\right) + \frac{\partial}{\partial z}\left(K_{Mv}\frac{\partial v}{\partial z}\right)$$

advection *Coriolis* *Pressure gradient* *Horizontal diffusion* *Vertical diffusion*

 # define UV_ADV

Tracer conservation

$$\frac{\partial T}{\partial t} + \vec{u}.\nabla T = \nabla_h\left(K_{Th}.\nabla_h T\right) + \frac{\partial}{\partial z}\left(K_{Tv}\frac{\partial T}{\partial z}\right)$$

$$\frac{\partial S}{\partial t} + \vec{u}.\nabla S = \nabla_h\left(K_{Sh}.\nabla_h S\right) + \frac{\partial}{\partial z}\left(K_{Sv}\frac{\partial S}{\partial z}\right)$$

#define TS_ADV

#define SOLVE3D (if #undef SOLVE3D) => Compuation of the depth integrated equations (Barotropic mode only).

If non-boussinesq (#define NBQ))
ifdef NBQ
define W_HADV_TVD
define W_VADV_TVD
endif

Available advection schemes

Equation	horizontal	vertical
3D momentum	UV_HADV_TVD UV_HADV_C2 **UV_HADV_UP3** UV_HADV_C4 UV_HADV_UP5 UV_HADV_C6 UV_HADV_WENO5	UV_VADV_TVD UV_VADV_C2 **UV_VADV_SPLINES** UV_VADV_WENO5
TRACERS	**TS_HADV_UP3** TS_HADV_RSUP3 TS_HADV_C4 TS_HADV_WENO5 TS_HADV_UP5 TS_HADV_C6 TS_HADV_RSUP5	TS_VADV_TVD TS_VADV_C2 TS_VADV_SPLINES **TS_VADV_AKIMA** TS_VADV_WENO5
2D momentum	**M2_HADV_UP3** M2_HADV_C2	

Advice !

Take default CPP options (Written in bold)

CROCO Equations and CPP options cppdefs.h //cppdefs_dev.h

CROCO
Coastal and Regional Ocean COmmunity model

Momentum conservation

$$\frac{\partial u}{\partial t} + \vec{u}.\nabla u - fv = -\frac{1}{\rho_0}\frac{\partial P}{\partial x} + \nabla_h\left(K_{Mh}.\nabla_h u\right) + \frac{\partial}{\partial z}\left(K_{Mv}\frac{\partial u}{\partial z}\right)$$

$$\frac{\partial v}{\partial t} + \vec{u}.\nabla v + fu = -\frac{1}{\rho_0}\frac{\partial P}{\partial y} + \nabla_h\left(K_{Mh}.\nabla_h v\right) + \frac{\partial}{\partial z}\left(K_{Mv}\frac{\partial v}{\partial z}\right)$$

advection *Coriolis* *Pressure gradient* *Horizontal diffusion* *Vertical diffusion*

```
# define UV_COR
```

Tracer conservation

$$\frac{\partial T}{\partial t} + \vec{u}.\nabla T = \nabla_h\left(K_{Th}.\nabla_h T\right) + \frac{\partial}{\partial z}\left(K_{Tv}\frac{\partial T}{\partial z}\right)$$

$$\frac{\partial S}{\partial t} + \vec{u}.\nabla S = \nabla_h\left(K_{Sh}.\nabla_h S\right) + \frac{\partial}{\partial z}\left(K_{Sv}\frac{\partial S}{\partial z}\right)$$

default is the Density Jacobian formulation with Cubic Polynomial fit from Shchepetkin et al. 2003. No cpp to activate in your cppdefs.h file. Advanced options are in cppdefs_dev.h

define UV_VIS2 // #define UV_VIS4
define TS_DIF2 //#define TS_DIF4

Horizontal mixing options

cppdefs.h //cppdefs_dev.h

CROCO
Coastal and Regional Ocean COmmunity model

Momentum conservation

$$\frac{\partial u}{\partial t} + \vec{u}.\nabla u - fv = -\frac{1}{\rho_0}\frac{\partial P}{\partial x} + \nabla_h\left(K_{Mh}.\nabla_h u\right) + \frac{\partial}{\partial z}\left(K_{Mv}\frac{\partial u}{\partial z}\right)$$

$$\frac{\partial v}{\partial t} + \vec{u}.\nabla v + fu = -\frac{1}{\rho_0}\frac{\partial P}{\partial y} + \nabla_h\left(K_{Mh}.\nabla_h v\right) + \frac{\partial}{\partial z}\left(K_{Mv}\frac{\partial v}{\partial z}\right)$$

advection Coriolis Pressure gradient Horizontal diffusion Vertical diffusion

define UV_VIS2 or #define UV_VIS4

if implicit dissipation in UV_HADV_UP3 is insufficient to handle subgrid-scale turbulence from strong shear currents .

Momentum horizontal mixing options

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

Horizontal tracers mixing options cppdefs.h //cppdefs_dev.h

Tracer conservation

$$\frac{\partial T}{\partial t} + \vec{u}.\nabla T = \nabla_h \left(K_{Th}.\nabla_h T\right) + \frac{\partial}{\partial z}\left(K_{Tv}\frac{\partial T}{\partial z}\right)$$

$$\frac{\partial S}{\partial t} + \vec{u}.\nabla S = \nabla_h \left(K_{Sh}.\nabla_h S\right) + \frac{\partial}{\partial z}\left(K_{Sv}\frac{\partial S}{\partial z}\right)$$

```
# define TS_VIS2 //
#define TS_VIS4
```

Horizontal mixing options are preselected in cppdefs_dev.h for compliance with advection options.

Tracers horizontal mixing options

TS_MIX_ISO	Activate mixing along isopycnal (isoneutral) surface
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)

cppdefs.h

```
#  undef  M3_FRC_BRY
#  define T_FRC_BRY
# endif
# undef  RVTK_DEBUG

#endif /* END OF CONFIGURATION CHOICE
*/

#include "cppdefs_dev.h"
#include "set_global_definitions.h"
```

Horizontal mixing options are preselected in cppdefs_dev.h for compliance with advection options.

! Don't change files: cppdefs_dev.h & set_global_definitions.h

File Edit View Projects Bookmarks Sessions Tools Settings Help

cppdefs.h | cppdefs_dev.h

```
#elif defined TS_HADV_RSUP5
#else
# define TS_HADV_UP3     /* 3rd-order upstream lateral advection */
# undef  TS_HADV_C4      /* 4th-order centered lateral advection */
# undef  TS_HADV_UP5     /* 5th-order upstream lateral advection */
# undef  TS_HADV_WENO5   /* 5th-order WENOZ    lateral advection */
# undef  TS_HADV_C6      /* 6th-order centered lateral advection */
# undef  TS_HADV_RSUP3   /* Rotated-Split UP3  lateral advection */
# undef  TS_HADV_RSUP5   /* Pseudo R-Split UP5 lateral advection */
#endif

/*
 Options for split-rotated advection-diffusion schemes
*/
#ifdef TS_HADV_RSUP3    /*  Rotated-Split 3rd-order scheme is:  */
# define TS_HADV_C4     /*    4th-order centered advection      */
# undef  TS_DIF2        /*               +                      */
# define TS_DIF4        /*         Hyperdiffusion  with         */
# undef  TS_MIX_GEO     /*        Geopotential rotation         */
# define TS_MIX_ISO     /*     or Isopycnal    rotation         */
#endif
#ifdef TS_HADV_RSUP5    /*    Pseudo RS 5th-order scheme is:    */
# define TS_HADV_C6     /*    6th-order centered advection      */
# undef  TS_DIF2        /*               +                      */
# define TS_DIF4        /*         Hyperdiffusion  with         */
# define TS_MIX_GEO     /*        Geopotential rotation         */
# undef  TS_MIX_ISO     /*     or Isopycnal    rotation         */
#endif
#if defined TS_HADV_C4 && !defined TS_HADV_RSUP3
                        /* 4th-order centered advection with:   */
# define TS_DIF2        /*   + Laplacian Diffusion              */
# undef  TS_DIF4        /*                                      */
# define TS_DIF_SMAGO   /*   + Smagorinsky diffusivity          */
# define TS_MIX_ISO     /*   + Isopycnal rotation               */
#endif

/*
  TS DIFFUSION: set default orientation
*/
#ifdef TS_MIX_S         /* Check if options are defined  */
#elif defined TS_MIX_GEO
#elif defined TS_MIX_ISO
#else
# define TS_MIX_S       /* Set iso-sigma diffusion as default */
#endif
/*
```

Vertical mixing options (1/3)

cppdefs.h

Momentum conservation

$$\frac{\partial u}{\partial t} + \vec{u}.\nabla u - fv = -\frac{1}{\rho_0}\frac{\partial P}{\partial x} + \nabla_h\left(K_{Mh}.\nabla_h u\right) + \frac{\partial}{\partial z}\left(K_{Mv}\frac{\partial u}{\partial z}\right)$$

$$\frac{\partial v}{\partial t} + \vec{u}.\nabla v + fu = -\frac{1}{\rho_0}\frac{\partial P}{\partial y} + \nabla_h\left(K_{Mh}.\nabla_h v\right) + \frac{\partial}{\partial z}\left(K_{Mv}\frac{\partial v}{\partial z}\right)$$

advection Coriolis Pressure gradient Horizontal diffusion Vertical diffusion

Tracer conservation

$$\frac{\partial T}{\partial t} + \vec{u}.\nabla T = \nabla_h\left(K_{Th}.\nabla_h T\right) + \frac{\partial}{\partial z}\left(K_{Tv}\frac{\partial T}{\partial z}\right)$$

$$\frac{\partial S}{\partial t} + \vec{u}.\nabla S = \nabla_h\left(K_{Sh}.\nabla_h S\right) + \frac{\partial}{\partial z}\left(K_{Sv}\frac{\partial S}{\partial z}\right)$$

PRONOSTIC AND NO PRONOSTIC SCHEMES OF VERTICAL MIXING ARE AVAILABLE

define GLS_MIXING or # define KPP_MIXING

Vertical mixing options (2/3)

cppdefs.h

Momentum conservation

$$\frac{\partial u}{\partial t} + \vec{u}.\nabla u - fv = -\frac{1}{\rho_0}\frac{\partial P}{\partial x} + \nabla_h(K_{Mh}.\nabla_h u) + \frac{\partial}{\partial z}\left(K_{Mv}\frac{\partial u}{\partial z}\right)$$

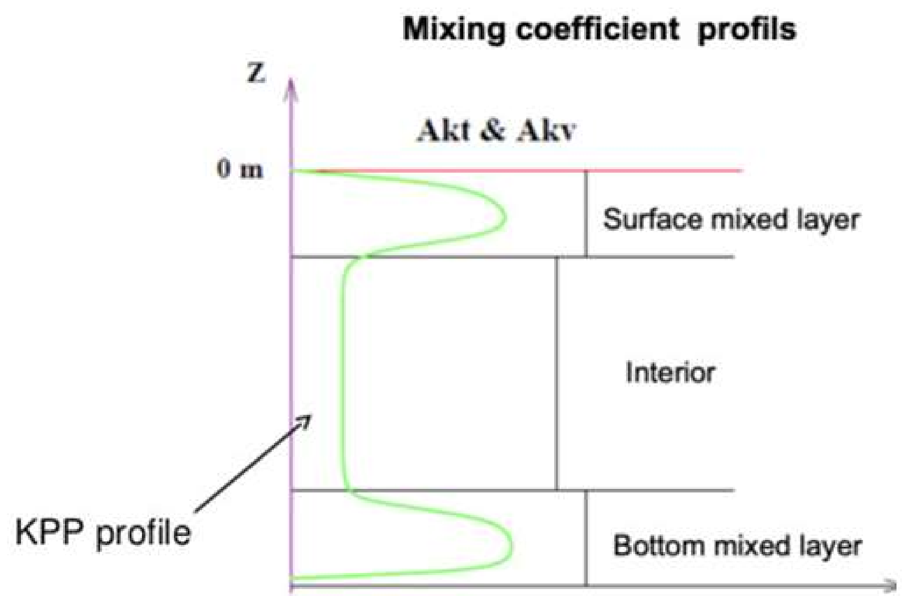
$$\frac{\partial v}{\partial t} + \vec{u}.\nabla v + fu = -\frac{1}{\rho_0}\frac{\partial P}{\partial y} + \nabla_h(K_{Mh}.\nabla_h v) + \frac{\partial}{\partial z}\left(K_{Mv}\frac{\partial v}{\partial z}\right)$$

advection Coriolis Pressure gradient Horizontal diffusion Vertical diffusion

Tracer conservation

$$\frac{\partial T}{\partial t} + \vec{u}.\nabla T = \nabla_h(K_{Th}.\nabla_h T) + \frac{\partial}{\partial z}\left(K_{Tv}\frac{\partial T}{\partial z}\right)$$

$$\frac{\partial S}{\partial t} + \vec{u}.\nabla S = \nabla_h(K_{Sh}.\nabla_h S) + \frac{\partial}{\partial z}\left(K_{Sv}\frac{\partial S}{\partial z}\right)$$



GLS mixing is based on two differntial equation one govering TKE ($q^2/2$) and the second governig ($q^2\ell$) where ℓ s the turbulent length scale.

Vertical mixing options (2/4) cppdefs.h

Momentum conservation

$$\frac{\partial u}{\partial t} + \vec{u}.\nabla u - fv = -\frac{1}{\rho_0}\frac{\partial P}{\partial x} + \nabla_h (K_{Mh}.\nabla_h u) + \frac{\partial}{\partial z}\left(K_{Mv}\frac{\partial u}{\partial z}\right)$$

$$\frac{\partial v}{\partial t} + \vec{u}.\nabla v + fu = -\frac{1}{\rho_0}\frac{\partial P}{\partial y} + \nabla_h (K_{Mh}.\nabla_h v) + \frac{\partial}{\partial z}\left(K_{Mv}\frac{\partial v}{\partial z}\right)$$

advection Coriolis Pressure gradient Horizontal diffusion Vertical diffusion

Tracer conservation

$$\frac{\partial T}{\partial t} + \vec{u}.\nabla T = \nabla_h (K_{Th}.\nabla_h T) + \frac{\partial}{\partial z}\left(K_{Tv}\frac{\partial T}{\partial z}\right)$$

$$\frac{\partial S}{\partial t} + \vec{u}.\nabla S = \nabla_h (K_{Sh}.\nabla_h S) + \frac{\partial}{\partial z}\left(K_{Sv}\frac{\partial S}{\partial z}\right)$$

No preselected options → The user should precise the convenient cpp option in his cppdefs.h

Available KPP options

LMD_MIXING	K-profile parametrisation
LMD_SKPP	Activate surface boundary layer KPP mixing
LMD_SKPP2005	Activate surface boundary layer KPP mixing (2005 version)
LMD_BKPP	Activate bottom boundary layer KPP mixing
LMD_BKPP2005	Activate bottom boundary layer KPP mixing (2005 version)
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
LMD_LANGMUIR	Activate Langmuir turbulence mixing

Available GLS options

GLS_MIXING	Activate Generic Length Scale scheme, default is k-epsilon (see below)
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]
CANUTO_A	Option for CANUTO A stability function (default, see below)
GibLau_78	Option for Gibson and Launder [1978] stability function
MelYam_82	Option for Mellor and Yamada [1982] stability function
KanCla_94	Option for Kantha and Clayson [1994] stability function
Luyten_96	Option for Luyten [1996] stability function
CANUTO_B	Option for CANUTO B stability function
Cheng_02	Option for Cheng *et al.* [2002] stability function

Vertical mixing options (3/3)

cppdefs.h

- KPP assumes that turbulence in the boundary layer is in equilibrium with surface and bottom fluxes => true for large scale models,
- for coastal applications, the scheme should respond to local forcing, respond rapidly to surface and bottom fluxes => GLS-type scheme preferred
- Analytical definition is also possible #define ANA_VMIX….

density equation

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

Needs to be specified in cppdefs.h

 # define SALINITY

 # define NONLIN_EOS

Open Boundaries (OBC's)

Two open boundaries : North and East

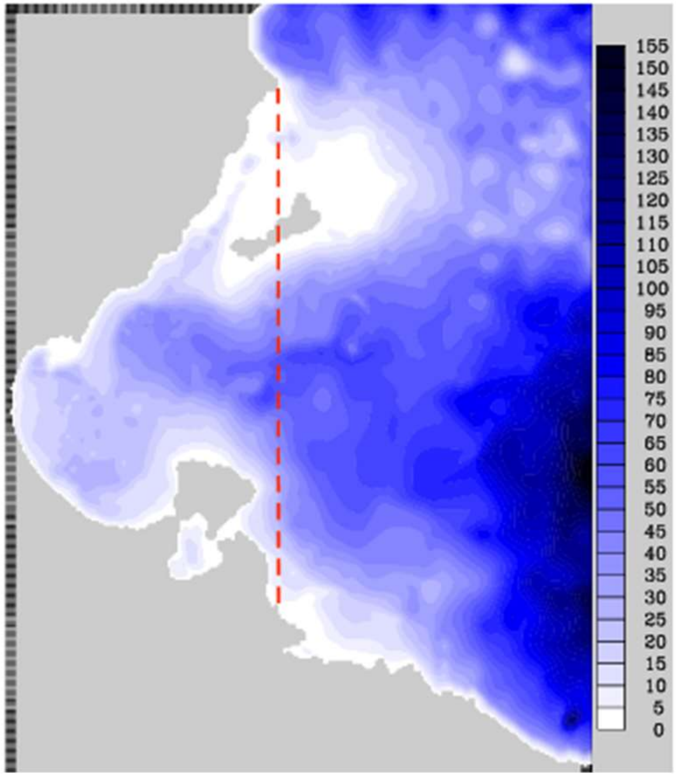
Two closed boundaries : South and West

In this case we have to activate in our cppdefs.h
#define OBC_NORTH
#define OBC_EAST

If we choose a domain delimited by the red dashed line, the only open boundary will be the Eastern one.

only activate :
#define OBC_EAST

Example : Golfe of Gabes



Bathymetry interpolated at 1/96°.
Depth are in meter.

OBC's CPP OPTIONS

* 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 OBC'S
* OBC_M2CHARACT is near as good and provides the best conditions for tidal forcing.

Preselected options in cppdefs_dev.h

OBC_M2CHARACT	**Activate OBCs from characteristic methods for barotropic velocities (default)**
OBC_M3ORLANSKI	**Activate radiative OBCs for baroclinic velocities (default)**
OBC_TORLANSKI	**Activate radiative OBCs for tracers (default)**

For **SSH** Chapman obc is set by default

Set your own choice in cppdefs.h if needed …

OBC_M2SPECIFIED	Activate specified OBCs for barotropic velocities
OBC_M2ORLANSKI	Activate radiative OBCs for barotropic velocities
OBC_VOLCONS	Enforce mass conservation at open boundaries (with OBC_M2ORLANSKI)
OBC_M3SPECIFIED	Activate specified OBCs for baroclinic velocities
OBC_TSPECIFIED	Activate specified OBCs for tracers
OBC_TUPWIND	Activate upwind OBCs for tracers

Surface forcing options (1)

cppdefs.h

Surface (z = ξ) boundary conditions

$$K_{Mv}\frac{\partial u}{\partial z} = \frac{\tau_s^x}{\rho_0}(x,y,t)$$

$$K_{Mv}\frac{\partial v}{\partial z} = \frac{\tau_s^y}{\rho_0}(x,y,t)$$

$$\kappa_{Tv}\frac{\partial T}{\partial z} = \frac{Q_T}{\rho_0 C_P} + \frac{1}{\rho_0 C_P}\frac{dQ_T}{dT}(T - T_{ref})$$

$$\kappa_{Sv}\frac{\partial S}{\partial z} = \frac{(E-P)S}{\rho_0}$$

Process to analytic forcing in croco by activating in your cppdefs.h analytical options:

```
# define NO_FRCFILE
# define ANA_SMFLUX
# define ANA_STFLUX..
```

Generate forcing file with **croco_pre-processing tools** (matlab //python) containing forcing variables: wind stress, heat , fresh water fluxes…

Use Bulk formulation by activating cpp options in your cppdefs (see next slide)

Surface forcing options (2)

cppdefs.h

CROCO
Coastal and Regional Ocean COmmunity model

Bulk formulation options

BULK_FLUX	Activate bulk formulation for surface turbulent fluxes (by default, COARE3p0 parametrizarion is used)
BULK_ECUMEV0	Use ECUMEv0 bulk formulation instead of COARE3p0 formulation
BULK_ECUMEV6	Use ECUMEv6 bulk formulation instead of COARE3p0 formulation
BULK_WASP	Use WASP bulk formulation instead of COARE3p0 formulation
BULK_GUSTINESS	Add in gustiness effect on surface wind module. Can be used for both bulk parametrizations.
BULK_LW	Add in long-wave radiation feedback from model SST

Additional functionalities...

SFLUX_CFB	Activate current feedback on ... (Renault et al., 2020)
CFB_STRESS	... surface stress (used by default when SFLUX_CFB is defined)
CFB_WIND_TRA	... surface tracers (used by default when SFLUX_CFB is defined)
SST_SKIN	Activate skin sst computation (Zeng & Beljaars, 2005)

QCORRECTION	Activate heat flux correction around model SST (if BULK_FLUX is undefined)
SFLX_CORR	Activate freshwater flux correction around model SSS (if BULK_FLUX is undefined)
ANA_DIURNAL_SW	Activate analytical diurnal modulation of short wave radiations (only appropriate if there is no diurnal cycle in data)

Bottom forcing options (2)

Croco.in + cppdefs.h

CROCO
Coastal and Regional Ocean COmmunity model

Bottom conditions for momentum (z =-H)

$$K_{Mv}\frac{\partial u}{\partial z} = \tau_b^x(x,y,t)$$

$$K_{Mv}\frac{\partial v}{\partial z} = \tau_b^y(x,y,t)$$

Needs to be specified in croco.in
Different formulations are available:

- Quadratic friction with log-layer $Z_{ob} \neq 0$
- Quadratic friction with $RDRG2 > 0$
- Linear friction with $RDRG > 0$

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
```

Additional cpp can be activated in your cppdefs.h

LIMIT_BSTRESS	Bottom stress limitation forstability
BSTRESS_FAST	Bottom stress computed in step3d_fast

Lateral forcing

Croco.in + cppdefs.h

CROCO
Coastal and Regional Ocean COmmunity model

Two ways for forcing on OBC's

Climaology strategy: Give a netcdf climatological file containing u,v,T,S,$\bar{u}$, $\bar{v}$, ξ on all points of the given domain. Path should be specified in croco.in

Boundary strategy: Give a netcdf boundary file containing u,v,T,S,$\bar{u}$, $\bar{v}$, ξ on OBC's. Path should be specified in croco.in

Add CPP options in your **cppdefs.h**

#define CLIMATOLOGY	Activate processing 2D/3D data (climatological or OGCM data..) used as forcing on OBC's and nudging layers
#define ZCLIMATOLOGY	Activate processing of sea level
#define M2CLIMATOLOGY	Activate processing of $\bar{u}, \bar{v}$
#define M3CLIMATOLOGY	Activate processing of baroclinic velocities
#define TCLIMATOLOGY	Activate processing of tracers

#define BRY_FRC	Activate forcing on OBC's points strictly
#define Z_FRC_BRY	Activate processing of sea level
#define M2_FRC_BRY	Activate processing of $\bar{u}, \bar{v}$
#define M3_FRC_BRY	Activate processing of baroclinic velocities
#define T_FRC_BRY	Activate processing of tracers

Useful for inter-annual forcing on high-resolution domains.

Tides forcing

Param.h+ cppdefs.h+croco.in

CROCO
Coastal and Regional Ocean COmmunity model

Tides elevations and currents are set on OBC's

$\xi_{tide}, \bar{u}_{tide}, \bar{v}_{tide}$ are added at the variables $\xi_{clim}, \bar{u}_{clim}, \bar{v}_{clim}$ at the open boundaries. For each tidal constituent, the user should provide the amplitude and phase of the elevation, as well as the ellipse parameters for the barotropic currents. These values can be obtained from a global tide model and interpolated onto your grid.

Indicate the number of tide constituents in **param.h**

Indicate the path of your tide forcing file in **croco.in**

Add CPP options in your **cppdefs.h**

```
#define TIDES
#ifdef TIDES
#define SSH_TIDES
#define UV_TIDES
#undef POT_TIDES
#undef TIDES_MAS
#ifnedf UV_TIDES
#define OBC_REDUCED_PHYSICS
#endif
#define TIDERAMP
#endif
#define OBC_M2CHARACT
```

CROCO

Coastal and Regional Ocean COmmunity model

RECOMMANDATIONS

Running

Coastal and Regional Ocean COmmunity model

*** Barotropic mode**

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

H is the maximum depth over your model domain
Δx grid horizontal resolution in m
g: gravity acceleration

for example for Hmax= 5km and Δx=30km we get $\Delta t \leq 85$ s

*** Baroclinic mode**

CFL condition for advection scheme : $\frac{\Delta t_{bc}}{\Delta x} V_{max} \leq \alpha^*$

Advection scheme	Max Courant number ($\alpha^\star$)
C2	1.587
UP3	0.871
SPLINES	0.916
C4	1.15
UP5	0.89
C6	1.00

Typical CFL values with croco time stepping algorithm

CFL :Courant–Friedrichs–Lewy (CFL) condition

In the case of UP3 advection scheme (default) with previous example (Hmax= 5km and Δx=30km)

If NDTFAST=$\frac{\Delta t_{bc}}{\Delta t_{bt}}$=60 ⟶ $\Delta t_{bc} \leq 5100\ s$

Then $V_{max} \leq 0.871 \frac{3\ 10^4}{5100} = 5.12\ m/s$: maximum allowed velocity

VERTCIAL GRID – Pressure gradient error pb

In s coordinates, the horizontal pressure gradient consists of two large terms that tend to cancel

$$-\left.\frac{\partial p}{\partial \xi}\right|_z = -g\rho|_{s=0} \cdot \frac{\partial \zeta}{\partial \xi} - g\int_s^0 \left[\frac{\partial z}{\partial s} \cdot \left.\frac{\partial \rho}{\partial \xi}\right|_s - \frac{\partial \rho}{\partial s} \cdot \left.\frac{\partial z}{\partial \xi}\right|_s\right] ds'$$

Interference of the discretization errors of these terms induces pressure gradient errors and drives spurious currents (in case of sharp topographic changes)

Recommendation: smooth the topography using a nonlinear filter and a criterium: $r = \Delta h / h < 0.2$ + choose high-order numerical schemes for advection-diffusion.

CROCO
Coastal and Regional Ocean COmmunity model

ANALYTICAL TEST CASE

BASIN cppdefs.h

```
/*
   This is "cppdefs.h": MODEL CONFIGURATION FILE
   ==== == ============ ===== ============= ====
*/
/*
        SELECT ACADEMIC TEST CASES
*/
#define  BASIN              /* Basin Example */
#undef  CANYON             /* Canyon Example */
```

See routines ana_grid.F, ana_initial.F, analytical.F

Closed open boundaries condition

```
#elif defined BASIN
/*
!                       Basin Example
!                       ===== =======
*/
# 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_FRCFILE
# undef  RVTK_DEBUG
```

BASIN **basin.in**

```
title:
    Basin Example

time_stepping: NTIMES   dt[sec]  NDTFAST  NINFO
                3240     9600     65       1

S-coord: THETA_S,    THETA_B,     Hc (m)
          1.0d0       0.0d0         10.d0

initial: NRREC  filename
          0
                          basin_rst.nc

restart:          NRST, NRPFRST / filename
                  20000000      -1
                          basin_rst.nc

history: LDEFHIS, NWRT, NRPFHIS / filename
            T      90        0
                          basin_his.nc
averages: NTSAVG, NAVG, NRPFAVG / filename
            1      3240      0
                          basin_avg.nc


primary_history_fields: zeta UBAR VBAR  U  V   wrtT(1:NT)
                         T    T    T     T  T   T
```

```
tracer_diff2: TNU2         [m^2/sec]
             1000. 0.

tracer_diff4: TNU4(1:NT)          [m^4/sec for all]
            30*0.d11

bodyforce:  levsfrc [level], levbfrc [level]
             10                1



diagnosticsM:    ldefdiaM   nwrtdiaM    nrpfdiaM /filename
                   T          270          0
                                  basin_diaM.nc

diagM_avg: ldefdiaM_avg  ntsdiaM_avg  nwrtdiaM_avg  nprfdiaM_avg /filename
               T             1          0             0
                                  basin_diaM_avg.nc

diagM_history_fields: diag_momentum(1:2)
                           T T

diagM_average_fields: diag_momentum_avg(1:2)
                           T T



diags_vrt:   ldefdiags_vrt, nwrtdiags_vrt, nrpfdiags_vrt /filename
                   T            0           0
                                  basin_diags_vrt.nc
diags_vrt_avg: ldefdiags_vrt_avg  ntsdiags_vrt_avg  nwrtdiags_vrt_avg
nprfdiags_vrt_avg /filename
               T            1          0           0
                                  basin_diags_vrt_avg.nc

diags_vrt_history_fields: diags_vrt
                           T
```

BASIN param.h, ana_grid.F

CROCO
Coastal and Regional Ocean COmmunity model

param.h

```
#if defined BASIN
      parameter (LLm0=60,   MMm0=50,   N=10)
```

ana_grid.F

```
# if defined BASIN
                    depth=5000.
                    f0=1.E-4
                    beta=2.E-11
!------------------------------------------------------------------
! Grid dimensions (Length_XI, Length_ETA)
!------------------------------------------------------------------
!
# define Length_XI_ISO_GRID Length_ETA*float(LLm0)/float(MMm0)
# define Length_ETA_ISO_GRID Length_XI*float(MMm0)/float(LLm0)
!
# ifdef AGRIF
      if (Agrif_Root()) then
# endif
# if defined BASIN
                    Length_XI =3600.0e+3
                    Length_ETA=2800.0e+3
```

From ana_grid.F :

$$\Delta x = \frac{3600\ 10^3}{60} = 60km, \Delta y = \frac{2800\ 10^3}{50} = 56\ km;\ H = Hmax = 5000m$$

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

$\longrightarrow$ $\Delta t_{bt} \leq 160$ s

croco.in

```
time_stepping: NTIMES    dt[sec]   NDTFAST
                3240      9600      65
```

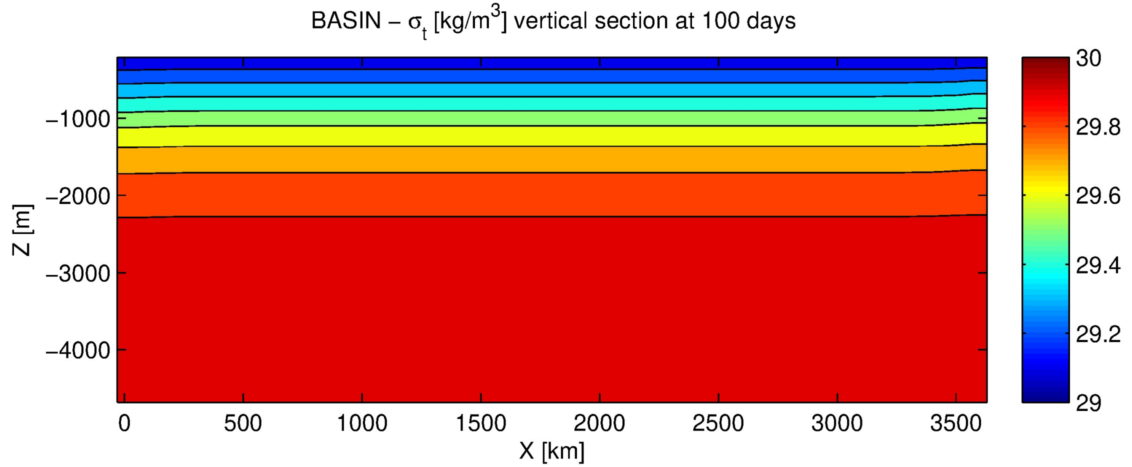
$$\Delta t_{bt} = \frac{9600}{65} = 147\ s$$

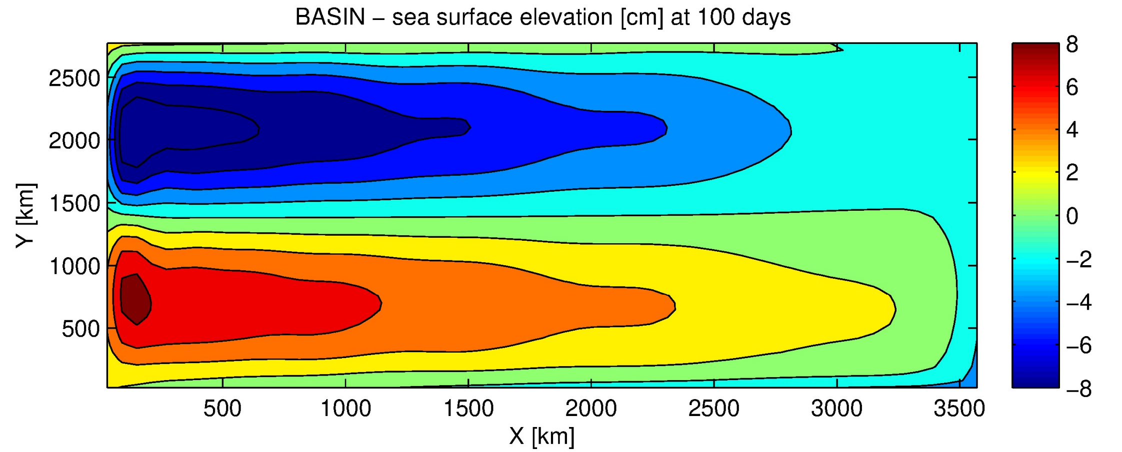
BASIN : Result visualizations

- Run plot_basin (matlab script in TEST_CASES directory)

Or

- Lunch ncview..





CROCO

Coastal and Regional Ocean COmmunity model

THANK YOU FOR YOUR ATTENTION