A python wrapper for the testing of Croco vertical closure schemes

1 Oceanic Boundary layer model

The 1D model for the horizontal velocity components $\mathbf{u}_h = (u, v)$ and for active tracers T and S defined on a domain such that $-H \leq z \leq 0$ reads

$$\begin{cases} \partial_t \mathbf{u}_h = -f\mathbf{e}_z \times (\mathbf{u}_h - \mathbf{u}_g) + \partial_z \left(K_m \partial_z \mathbf{u}_h\right) - \mathcal{P}_x \mathbf{e}_x \\ \partial_t T = \partial_z \left(K_s \partial_z T - \hat{\gamma}\right) + \dfrac{1}{\rho_0 C_p}\left[-\partial_z Q_s\right] + \delta(T - T_{\mathrm{LS}}) \\ \partial_t S = \partial_z \left(K_s \partial_z S - \hat{\gamma}\right) + \delta(S - S_{\mathrm{LS}}) \end{cases} \tag{1.1}$$

where K_m and K_s are respectively the turbulent viscosity and diffusivity, $\mathbf{e}_z$ a unit vector in the vertical direction, f is the Coriolis frequency, $\mathbf{u}_g$ represents the geostrophic current, $\mathcal{P}_x$ is a forcing term in x-direction, $\hat{\gamma}$ is a countergradient flux (see below), δ is a restoring time towards large scale temperature T_{LS} and salinity S_{LS}, and Q_s is a downward solar flux. This model must be completed by top and bottom boundary conditions, for momentum

$$K_m \partial_z \mathbf{u}_h(t)|_{z=0} = \boldsymbol{\tau}(t)/\rho_0, \qquad K_m \partial_z \mathbf{u}_h(t)|_{z=-H} = r_D \mathbf{u}_h(z=-H,t), \qquad r_D = C_D^b \|\mathbf{u}_h(z=-H,t)\|$$

and tracers (with Q_0 the net heat flux and $Q_s^{\downarrow}$ the surface downward solar radiation)

$$K_s \partial_z T(t)|_{z=0} = -\frac{1}{\rho_0 C_p}\left(Q_0(t) - Q_s^{\downarrow}\right), \qquad K_s \partial_z S(t)|_{z=0} = -S(z=0,t)(E-P), \qquad Q_s(t)|_{z=0} = Q_s^{\downarrow}$$

as well as

$$\partial_z T(t)|_{z=-H} = \Gamma_T, \qquad \partial_z S(t)|_{z=-H} = \Gamma_S.$$

Those boundary conditions are written in a generic way to allow different treatment of the bottom boundary condition at $z = -H$:

- Choosing $r_D \neq 0$ and $\Gamma_T = \Gamma_S = 0$ amounts to consider that $z = -H$ corresponds to the oceanic bottom.
- Choosing $r_D = 0$, $\Gamma_T = \partial_z T(t=0)|_{z=-H}$, and $\Gamma_S = \partial_z S(t=0)|_{z=-H}$ amounts to consider that the equations are solved on an upper domain of a deep ocean cut at $z = -H$.

In case $z = -H$ corresponds to a rough surface, the bottom roughness is defined through the parameter r_D. The vertical penetration of the surface solar radiation flux $Q_s^{\downarrow}$ is given by a standard Jerlov law (consistent with the croco parameterization). The 1D model also requires initial conditions which will be defined latter.

In (1.1), the turbulent viscosity/diffusivity and the non-local term $\hat{\gamma}$ are given either using a 0-equation closure scheme (KPP, Shchepetkin, 2005; McWilliams et al., 2009) or a 2-equation closure scheme (GLS, Burchard et al., 1998; Umlauf and Burchard, 2003, 2005) whose implementations are taken from the Croco source code (see lmd_skpp1994.F, lmd_skpp2005.F and gls_mix2017.F subroutines and description for GLS in appendix A). Those closure schemes both require the computation of the Brunt-Vaisala frequency which can be done either using a standard nonlinear equation of state for sea-water

$$\rho = r_0 + T \times P_3(T) + S \times P_3'(T) + \sqrt{S} \times P_2(T) + r_1 S\sqrt{S}$$

where P_3 and $P_3^{'}$ are thrid-order polynomials and P_2 a second-order polynomial, or a linear equation of state depending on temperature only (with T_0 a temperature of reference and α the thermal expansion coefficient)

$$\rho = -\alpha\rho_0(T - T_0)$$

2 Elements on the temporal and spatial discretization of the 1D model

A fortran code has been developed to integrate the 1D model from a given time-step n to the following time-step $n+1$.

Time-integration of the Coriolis term

The Coriolis term (including the geostrophic guide) is integrated using a standard θ scheme,

$$u^{n+1} = \left(\frac{1-\theta(1-\theta)(f\Delta t)^2}{1+\theta^2(f\Delta t)^2}\right)u^n + \left(\frac{(f\Delta t)}{1+\theta^2(f\Delta t)^2}\right)(v^n - v_g) + \left(\frac{\theta(f\Delta t)^2}{1+\theta^2(f\Delta t)^2}\right)u_g$$

$$v^{n+1} = \left(\frac{1-\theta(1-\theta)(f\Delta t)^2}{1+\theta^2(f\Delta t)^2}\right)v^n - \left(\frac{(f\Delta t)}{1+\theta^2(f\Delta t)^2}\right)(u^n - u_g) + \left(\frac{\theta(f\Delta t)^2}{1+\theta^2(f\Delta t)^2}\right)v_g$$

the default value in the code is $\theta = 0.55$.

Time-integration of vertical viscosity

An other point that is worth a description is the time-integration of the viscous term $\partial_z\left(K_m\partial_z\mathbf{u}_h\right)$. For this term, the surface boundary condition is treated in an explicit way (subject to a stability constraint if a bulk formulae is used, Beljaars et al., 2017) while the bottom boundary condition and the diffusion operator are treated implicitly. For a given velocity component u and diffusion coefficient K, the corresponding discretization is thus

$$u_k^{n+1} - u_k^n = \frac{\Delta t}{\Delta z_k}\left[\frac{K_{k+1/2}}{\Delta z_{k+1/2}}(u_{k+1}^{n+1} - u_k^{n+1}) - \frac{K_{k-1/2}}{\Delta z_{k-1/2}}(u_k^{n+1} - u_{k-1}^{n+1})\right], \qquad 2 \leq k \leq N-1$$

$$u_N^{n+1} - u_N^n = \frac{\Delta t}{\Delta z_N}\left[(\tau_x^n/\rho_0) - \frac{K_{N-1/2}}{\Delta z_{N-1/2}}(u_N^{n+1} - u_{N-1}^{n+1})\right]$$

$$u_1^{n+1} - u_1^n = \frac{\Delta t}{\Delta z_1}\left[\frac{K_{3/2}}{\Delta z_{3/2}}(u_2^{n+1} - u_1^{n+1}) - r_D u_1^{n+1}\right]$$

with N the total number of vertical levels ($k = N$ at the surface and $k = 1$ at the bottom). The resulting tridiagonal system of linear algebraic equations is solved using the Thomas tridiagonal algorithm. The parameter r_D is given as an input to the fortran code and can be computed using different type of parameterization. An example is the formulation proposed by Shchepetkin (2009)

$$r_D = \|(\mathbf{u}_h)_1\|\left(\frac{\kappa}{\left(\frac{z_0}{\Delta z_1}+1\right)\ln\left(\frac{\Delta z_1}{z_0}+1\right)-1}\right)^2 \tag{2.1}$$

with z_0 a roughness length. In turbulence closure schemes it is assumed that

$$(u_\star^s)^2 = \|\boldsymbol{\tau}\|/\rho_0, \qquad (u_\star^b)^2 = r_D\|(\mathbf{u}_h)_1\|$$

3 Driving the 1D model from Python script

As mentioned above, the fortran code performs the integration of the 1D model over 1 time-step. This code can be called from a Python script. In this case, in the Python script the following steps must be done

- Initialization of the vertical grid, i.e. of the arrays z_r, z_w and Hz
- Initialization of the large-scale data $\mathbf{u}_G$, $T_{\rm LS}$ and $S_{\rm LS}$ as well as the restoring coefficient δ.
- Initialization of prognostic variables $\mathbf{u}_h$, T, and S (as well as TKE and ψ if needed)
- Initialization of the Γ_T and Γ_S values for the bottom boundary condition
- Computation of r_D (at each time-step)
- Computation of surface fluxes $\boldsymbol{\tau}$, Q_0, $Q_s^{\downarrow}$ and $S(E-P)$ (at each time-step if time-dependent)
- Initialization of various model parameters : Ri_c, ρ_0, α, $\mathcal{P}_x$, ...

Assuming that all those steps have been properly done, a typical integration loop over $\mathrm{nb\_steps}$ time steps in the Python script would look like (assuming constant in time surface fluxes)

```
for kt in range(nb_steps):

    nstp = 1 +  kt    % 2
    nnew = 1 + (kt+1) % 2
    time = dt*float(kt)       # time in seconds

    # bottom drag (if needed)
    u1       = uoce[0,nstp-1]
    v1       = voce[0,nstp-1]
    cff       = np.sqrt( u1*u1 + v1*v1 )
    r_D      = cff*pow(vonKar/((1.+Zob/Hz[0])*np.log(1.+Hz[0]/Zob) -1.),2)

    scm_oce.obl_stp(z_r,z_w,Hz,                                  \
                    unudge,vnudge,tnudge,                        \
                    uoce,voce,toce,turb,                        \
                    rho0,rho1,Akv,Akt,r_D,     \
                    sustr,svstr,srflx,heatflx,freshflx,    \
                    dTdz_bot,delta,fcor,Ric,hbls,dt,dpdx,    \
                    turbulence_scheme,stability_function,     \
                    lin_eos,alpha,T0,Zob,Neu_bot,       \
                    nstp,nnew,nz,ntra,nt,ngls )
```

where $\mathrm{scm\_oce}$ is the name of the fortran module containing the $\mathrm{obl\_stp}$ subroutine. The meaning and units of various arguments of the $\mathrm{obl\_stp}$ subroutine are given in table 3. Thanks to this python wrapper it is possible to define a single Python script to define a given testcase and to run this testcase with various input parameters while doing the diagnostics online. In the following we evaluate the current Croco implementations of KPP and GLS for a set of scenarios well documented in the literature (e.g. Duhaut, 2009).

Python name(dimension)	Units	Staggering	Description
z_r(N)	m	ρ	depth of cell centers
z_w(N + 1)	m	w	depth of cell interfaces
Hz(N)	m	ρ	vertical resolution Δz
unudge(N)/vnudge(N)	m s^{-1}	ρ	geostrophic currents
tnudge(N, ntra)	^{o}C or psu	ρ	large-scale temperature/salinity
uoce(N, nt)/voce(N, nt)	m s^{-1}	ρ	prognostic horizontal velocity components
toce(N, nt, ntra)	^{o}C or psu	ρ	prognostic temperature/salinity
turb(N + 1, nt, ngls)	m^2 s^{-2}	w	GLS prognostic variables
ρ_0	kg m^{-3}		reference density
$\rho_1(N)$	kg m^{-3}	ρ	density perturbation
Akv(N + 1)	m^2 s^{-1}	w	turbulent viscosity
Akt(N + 1, ntra)	m^2 s^{-1}	w	turbulent diffusivity
r_D	m s^{-1}		bottom drag intensity
sustr/svstr	m^2 s^{-2}		surface wind-stress
srflx	^{o}C m s^{-1}		downward solar radiation
heatflx	^{o}C m s^{-1}		net heat flux
freshflx	psu m s^{-1}		net freshwater flux
dTdz_bot($ntra$)	^{o}C m^{-1} or psu m^{-1}		Bottom boundary condition for tracers
delta(N)	s^{-1}	ρ	vertical profile of inverse restoring time
fcor	s^{-1}		Coriolis frequency
Ric			Critical Richardson number (KPP only)
hbls	m		Boundary layer depth (KPP only)
dt	s		time-step
dpdx	m s^{-2}		constant forcing term for u-equation
turbulence_scheme			=0 KPP; =1 K-ω; =2 K-ϵ; =3 Gen
stability_function			=0 Canuto A; =1 Gibson-Launder 78;
			=2 Mellor-Yamada 82; =3 Kantha-Clayson 94;
			=4 Luyten 96; =5 Canuto B; =6 Cheng 02
lin_eos			boolean for equation of state
alpha, T0	^{o}C^{-1}, ^{o}C		eos parameters if lin_eos= True
Zob	m		bottom roughness length
Neu_bot			boolean for the bottom B.C. for GLS variables

Table 1
Description of the arguments of the obl_stp subroutine. Typical value for ntra (number of tracers), ngls (number of GLS prognostic variables), and nt (number of time-step stored) is 2.

4 Idealised scenarios

4.1 Wind-induced deepening of boundary layer (Kato & Phillips case)

Parameter values are the following :

H	50 m	ρ_0	1024 kg m^{-3}	$\mathcal{P}$	0 m s^{-2}
N	100 levels	f	0 s^{-1}	δ	0 s^{-1}
Δt	30 s	T	30 hours	Neu_bot	True
lin_eos	True	T_0	16 ^{o}C	α	2×10^{-4} ^{o}C^{-1}
r_D	0	$z_{0,b}$	0		

The large-scale forcing (i.e. $\mathbf{u}_g$, $T_{\rm LS}$, $S_{\rm LS}$) is set to zero and the initial conditions are

$$S(z,t=0) = 35 \text{ psu}, \quad T(z,t=0) = T_0 - N_0^2(\alpha g)^{-1}|z|, \quad u(z,t=0) = v(z,t=0) = 0 \text{ m s}^{-1}$$

with $N_0 = 0.01$ s^{-1} and $g = 9.81$ m s^{-2}. Bottom boundary conditions are

$$r_D(t) = 0 \text{ m s}^{-1}, \quad \Gamma_T(t) = -N_0^2(\alpha g)^{-1}, \quad \Gamma_S(t) = 0$$

and surface forcing (with $u_\star^s = 0.01$ m s^{-1})

$$\tau_x(t)/\rho_0 = (u_\star^s)^2, \quad \tau_y = 0, \quad Q_0(t) = 0, \quad Q_s^{\downarrow}(t) = 0, \quad S(E-P)(t) = 0.$$

The relevant parameter to evaluate the simulations is the temporal evolution of mixed layer depth (defined here as the depth where the turbulent viscosity first reaches its background value, starting from the surface) which has been empirically shown to behave as

$$D_{\rm ml}(t) = 1.05 u_\star^s \sqrt{t/N_0}$$

Comparisons between this empirical relation and the results from model simulations are shown in Fig. 1, as well as the final temperature and turbulent viscosity profiles at final time in Fig. 2.

4.2 Free-convection simulation with constant cooling (Willis & Deardorff)

Parameter values are the following :

H	50 m	ρ_0	1024 kg m^{-3}	$\mathcal{P}$	0 m s^{-2}
N	50 levels	f	0 s^{-1}	δ	0 s^{-1}
Δt	30 s	T	72 hours	Neu_bot	True
lin_eos	False	T_0	(not needed)	α	(not needed)
r_D	0	$z_{0,b}$	0		

Again, the large-scale forcing (i.e. $\mathbf{u}_g$, $T_{\rm LS}$, $S_{\rm LS}$) is set to zero and the initial conditions are

$$S(z,t=0) = 35 \text{ psu}, \quad T(z,t=0) = T_0 - N_0^2(\alpha g)^{-1}|z|, \quad u(z,t=0) = v(z,t=0) = 0 \text{ m s}^{-1}$$

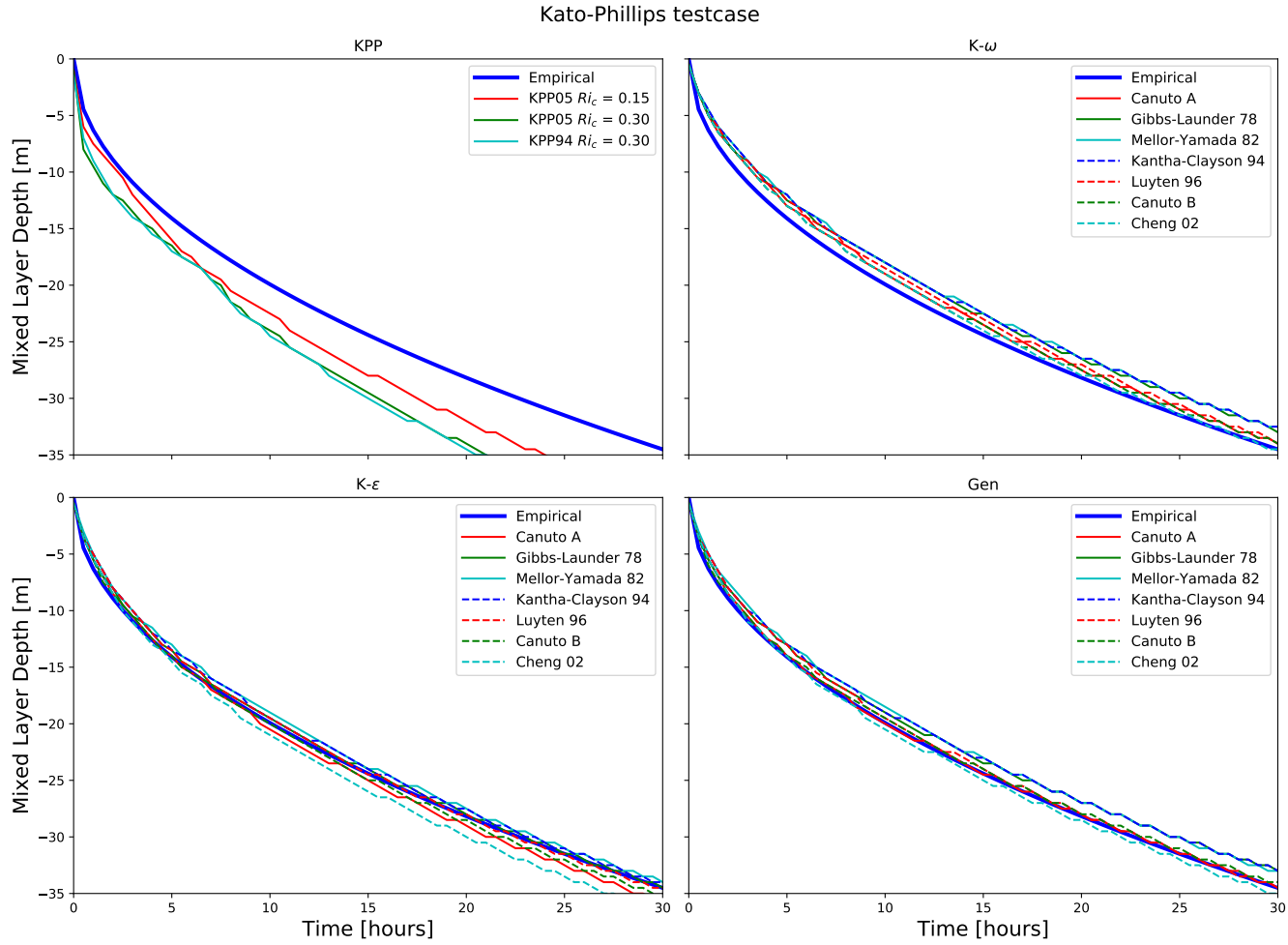

Fig. 1. Evolution of mixed layer depth with time for the Kato-Phillips case for different turbulent closure schemes: KPP (top,left) with different values of Ri_c and different versions of the scheme; $\mathrm{K}-\omega$ (top right) for different choices of stability functions; $\mathrm{K}-\epsilon$ (bottom left); Gen (bottom right). The empirical function $D_{\mathrm{ml}}(t)$ is shown with a blue solid line.

with $N_0 = \sqrt{\alpha g \times (0.1\ {}^0\mathrm{C\ m^{-1}})}$ and $g = 9.81\ \mathrm{m\ s^{-2}}$. Bottom boundary conditions are

$$r_D(t) = 0\ \mathrm{m\ s^{-1}}, \quad \Gamma_T(t) = -N_0^2(\alpha g)^{-1}, \quad \Gamma_S(t) = 0$$

and surface forcing

$$\tau_x(t) = \tau_y(t) = 0, \quad Q_0(t) = -\frac{100}{\rho_0 C_p}, \quad Q_s^{\downarrow}(t) = 0., \quad S(E-P)(t) = 0.$$

The relevant parameter to evaluate the simulations is the entrainment depth after $3\ \mathrm{days}$ which should be around $11;\mathrm{m}$. Results are shown in Fig. 3 where it is clear that 2-equation models give results with good comparison with existing LES simulations.

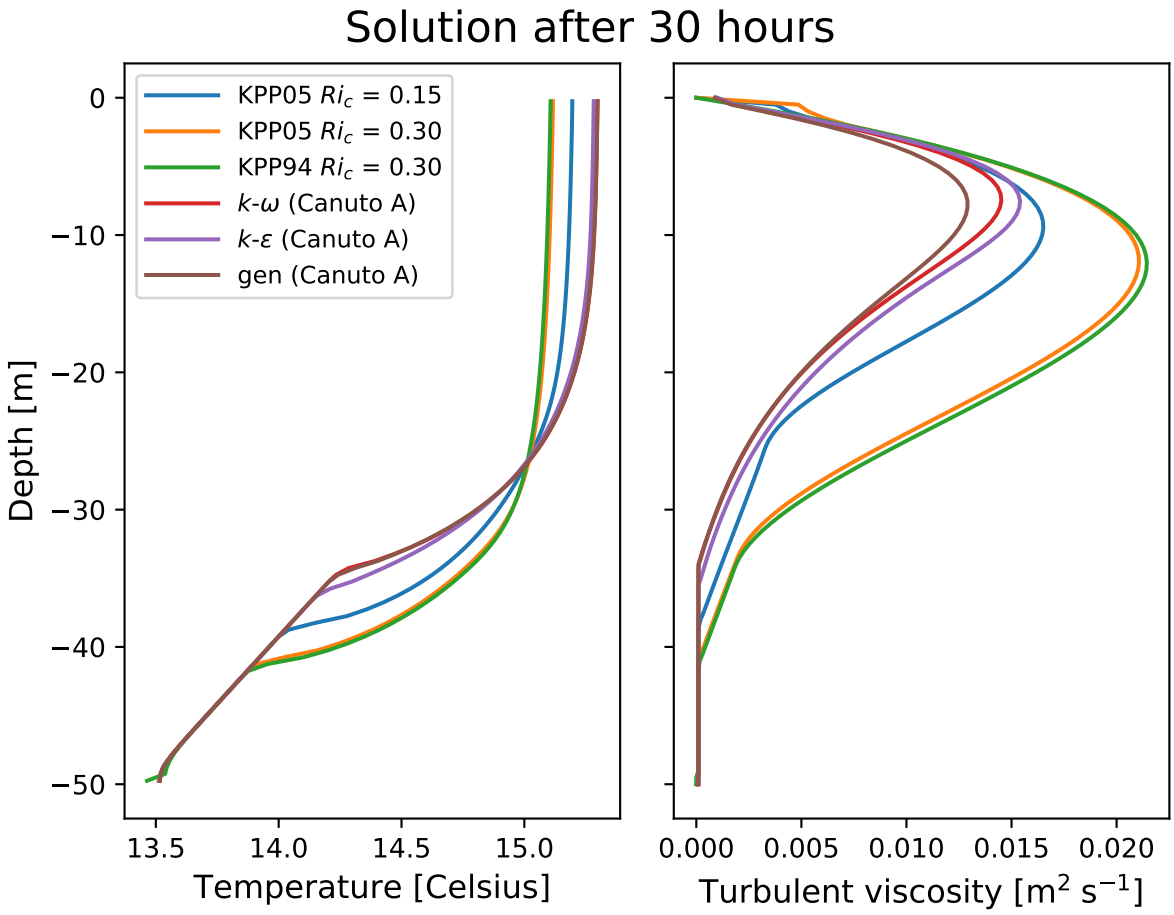

Fig. 2. Final state after $t = 30$h for the Kato-Phillips case: temperature in Celsius (left) and turbulent viscosity profile (right) for several parameterization schemes (see legend).

4.3 Pressure-gradient driven flow

H	5 m	ρ_0	1024 kg m^{-3}	$\mathcal{P}$	$(2 \times 10^{-6})g$
N	50 levels	f	0 s^{-1}	δ	0 s^{-1}
Δt	30 s	T	60 hours	Neu_bot	True or False
lin_eos	True	T_0	16 ^{0}C	α	2×10^{-4} ^{o}C^{-1}
r_D	see eqn (2.1)	$z_{0,b}$	0.01 m		

Again, the large-scale forcing (i.e. $\mathbf{u}_g$, $T_{\rm LS}$, $S_{\rm LS}$) is set to zero and the initial conditions are

$$S(z, t = 0) = 35 \text{ psu}, \quad T(z, t = 0) = 16\ ^0\text{C}, \quad u(z, t = 0) = v(z, t = 0) = 0 \text{ m s}^{-1}$$

Bottom boundary conditions are

$$r_D(t) = \|(\mathbf{u}_h)_1\| \left(\frac{\kappa}{\left(\frac{z_{0,b}}{\Delta z_1} + 1\right) \ln\left(\frac{\Delta z_1}{z_{0,b}} + 1\right) - 1} \right)^2, \quad \Gamma_T(t) = 0, \quad \Gamma_S(t) = 0$$

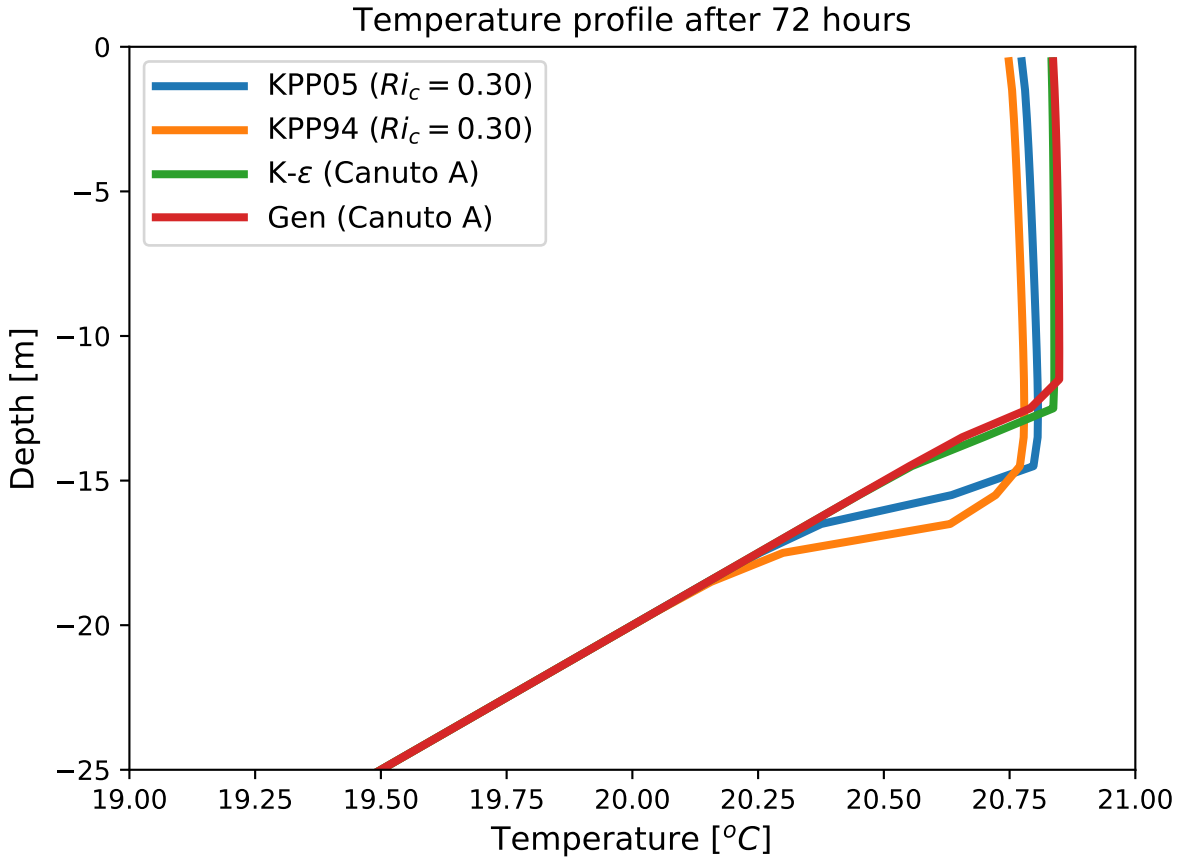

Fig. 3. Final state after $t = 72\mathrm{h}$ for the Willis-Deardorff case: temperature in Celsius for several parameterization schemes (see legend). From LES simulations, the depth of the mixed-layer should be around 11 m.

and surface forcing

$$\tau_x(t) = \tau_y(t) = 0, \quad Q_0(t) = 0, \quad Q_s^{\downarrow}(t) = 0., \quad S(E-P)(t) = 0.$$

All simulations have reached a steady state after 60 hours. For this test-case, it is possible to anticipate how the steady state solution should look like. Indeed, only the zonal velocity component is nonzero and its evolution is given by

$$\partial_t u = \partial_z(K_m \partial_z u) - \mathcal{P}_x$$

using the stress defined as $(\tau_x/\rho_0) = K_m \partial_z u$, when the solution has reached its stationary limit, the solution satisfies

$$\frac{\partial_z \tau_x}{\rho_0} = \mathcal{P}_x \qquad \rightarrow \qquad \tau_x/\rho_0 = \mathcal{P}_x z$$

in particular $\mathcal{P}_x z = u_{\star,b}^2$ at $z = -H$. When approaching the bottom the solution should behave like

$$u(z) = \frac{u_{\star,b}}{\kappa} \ln\left(\frac{z + H + z_{0,b}}{z_{0,b}}\right)$$

Results and comparison to this analytical profile are shown in Fig. 4.

4.4 Turbulent Ekman bottom boundary layer (Andren et al., 1994)

This first test is a standard test from the atmospheric community that can be transposed easily to the oceanic context

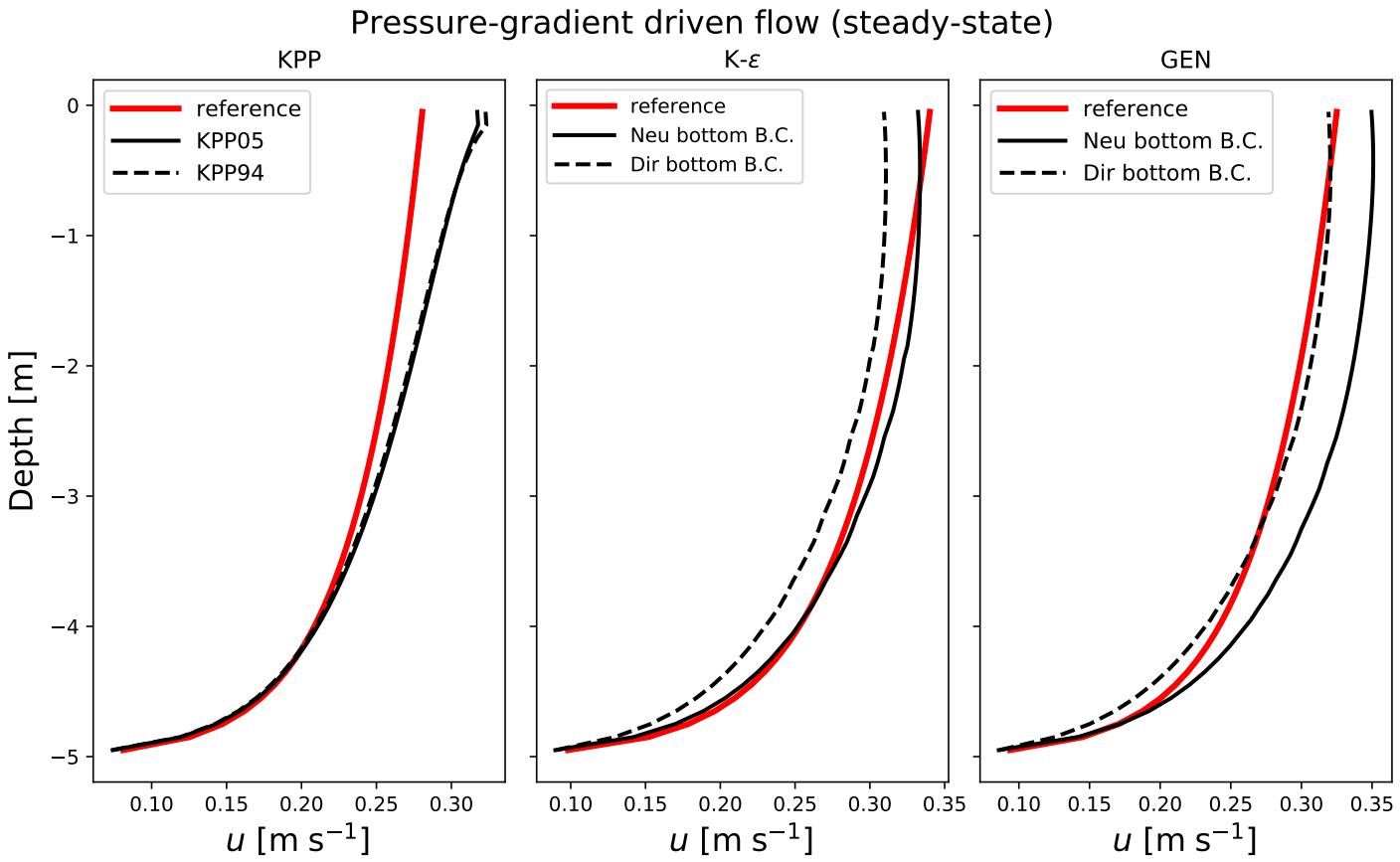

Fig. 4. Stationary solution for the pressure-gradient driven flow for different closure schemes and bottom boundary conditions. The thick red line corresponds to the solution of reference when approaching the bottom. Since each turbulent scheme will produce different values of $u_{\star,b}$ it explains why solutions of reference are not identical among different options.

H	1500 m	ρ_0	1.22 kg m^{-3}	$\mathcal{P}$	0 m s^{-2}
N	40 levels	f	10^{-4} s^{-1}	δ	0 s^{-1}
Δt	30 s	T	24 hours	Neu_bot	True or False
lin_eos	True	T_0	16 ^{0}C	α	2×10^{-4} ^{o}C^{-1}
r_D	see eqn (2.1)	$z_{0,b}$	0.1 m		

For this testcase a geostrophic velocity is imposed

$$u_G = 10 \text{ m s}^{-1}, \qquad v_G = 0 \text{ m s}^{-1}$$

and used to initialise the model velocities. All other model parameters are set to zero or ignored. For this testcase a solution of reference given by LES experiments exists and is shown in Fig. 5 (bottom 2 panels). Results obtained with our 1D model are shown on the same figure (top 3 panels).

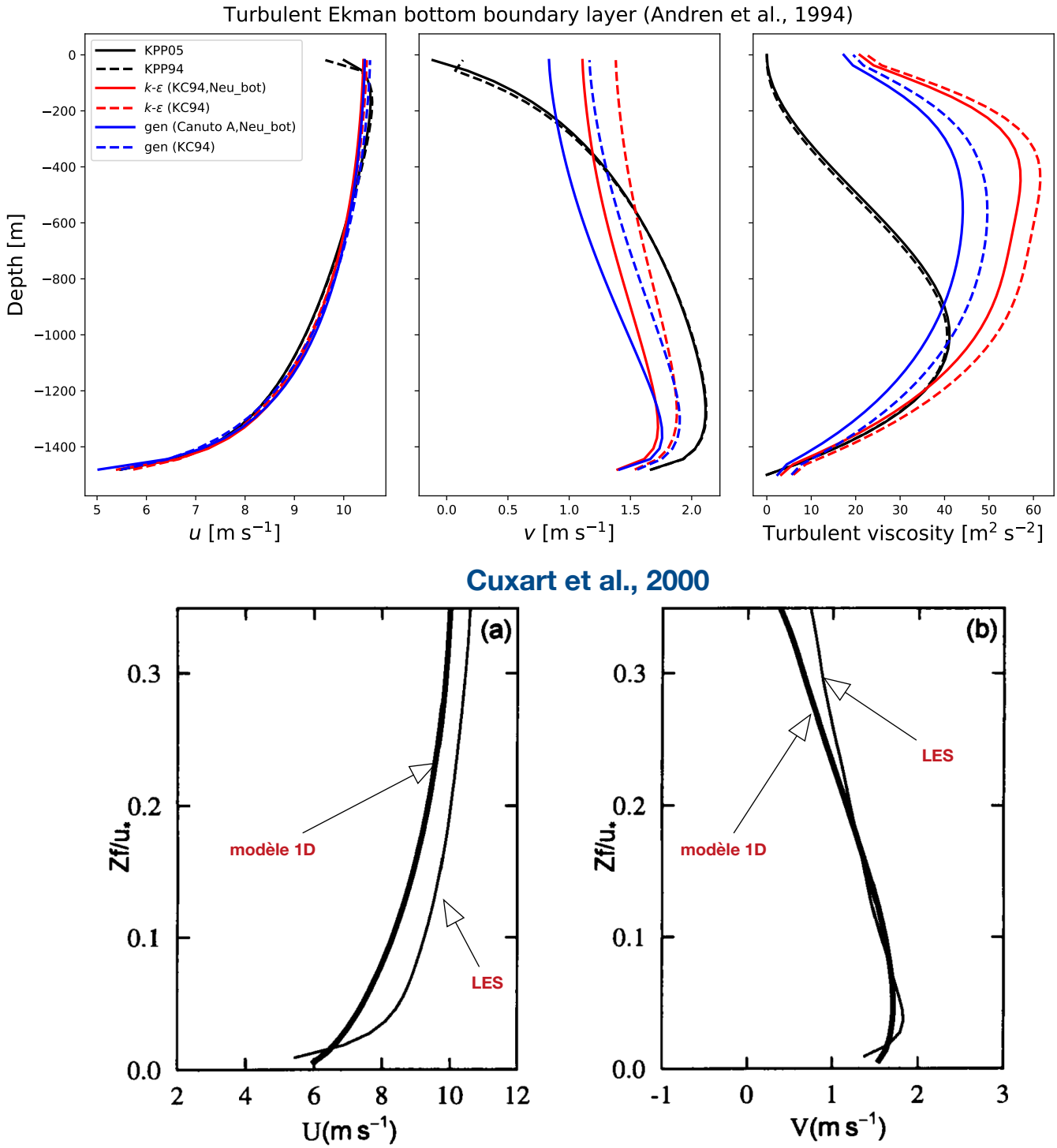

Fig. 5. Simulation of a turbulent Ekman bottom boundary layer for different closure schemes (top 3 panels). Solution of reference obtained through LES simulations (bottom 2 panels) and solution obtained with the CBR00 scheme (thick black line on bottom panels). The vertical axis in bottom panels is rescaled by a constant value $f/u_\star$.

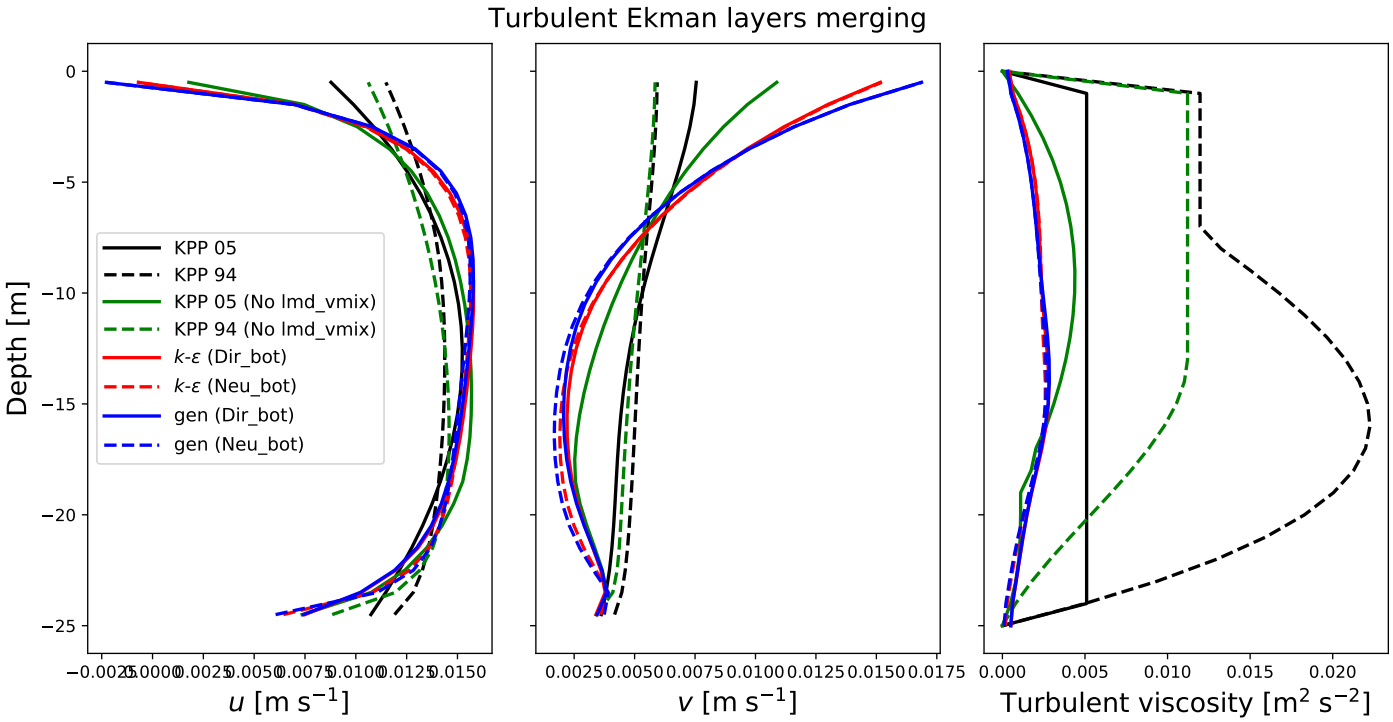

Fig. 6. Simulation of a turbulent Ekman boundary layer with merging of the surface and bottom BLs for different closure schemes (see legend).

4.5 Merging of turbulent Ekman boundary layers

H	25 m	ρ_0	1024 kg m^{-3}	$\mathcal{P}$	0 m s^{-2}
N	25 levels	f	10^{-4} s^{-1}	δ	0 s^{-1}
Δt	30 s	T	24 hours	Neu_bot	True or False
lin_eos	True	T_0	16 ^{0}C	α	2×10^{-4} ^{o}C^{-1}
r_D	see eqn (2.1)	$z_{0,b}$	0.05 m		

For this testcase a geostrophic current is imposed

$$u_G = 0.015 \text{ m s}^{-1}, \qquad v_G = 0 \text{ m s}^{-1}$$

and used to initialise the model velocities. Salinity and temperature are held constant and the surface boundary condition for momentum is

$$\tau_x/\rho_0 = -(u^s_\star)^2, \quad \tau_y/\rho_0 = 0, \quad u^s_\star = 2.5 \times 10^{-3} \text{ m s}^{-1}.$$

Results are shown in Fig. 6.

A Equivalence between Fortran variable names and the discrete equations

u_k^n	u(k,nstp)	v_k^n	v(k,nstp)
u_k^{n+1}	u(k,nnew)	v_k^{n+1}	v(k,nnew)
T_k^n	t(k,nstp,1)	S_k^n	t(k,nstp,2)
T_k^{n+1}	t(k,nnew,1)	S_k^{n+1}	t(k,nnew,2)
ρ^n	rho1(k)	$(N^2)_{k+1/2}^n$	bvf(k)
$(K_m)_{k+1/2}$	Akv	$(K_s)_{k+1/2}$	Akt
$\mathrm{TKE}_{k+1/2}^n$	trb(k,nstp,1)	$\psi_{k+1/2}^n$	trb(k,nstp,2)
$\mathrm{TKE}_{k+1/2}^{n+1}$	trb(k,nnew,1)	$\psi_{k+1/2}^{n+1}$	trb(k,nnew,2)
$(u_g)_k$	unudge(k)	$(v_g)_k$	vnudge(k)
$(T_{\mathrm{LS}})_k$	tnudge(k,1)	$(S_{\mathrm{LS}})_k$	tnudge(k,2)
z_k	z_r(k)	$z_{k+1/2}$	z_w(k)
Δz_k	Hz(k)	ρ_0	rho0
r_D	r_D	$\frac{Q_s^{\downarrow}}{\rho_0 C_p}$	srflx
τ_x/ρ_0	sustr	τ_y/ρ_0	svstr
$\frac{Q_0}{\rho_0 C_p}$	stflx1	$S(E-P)$	stflx2
δ_k	delta(k)	f	f
Γ_T	dtdz_bot(1)	Γ_S	dtdz_bot(2)
Ri_c	Ricr	h_{bls}	hbls
β	rdamp	α	alpha
T_0	T0	$\hat{\gamma}_{k+1/2}$	ghat(k)

References

Andren, A., Brown, A.R., Mason, P.J., Graf, J., Schumann, U., Moeng, C.H., Nieuwstadt, F.T.M., 1994. Large-eddy simulation of a neutrally stratified boundary layer: A comparison of four computer codes. Quart. J. Roy. Meteorol. Soc. 120(520), 1457–1484.

Beljaars, A., Dutra, E., Balsamo, G., Lemarié, F., 2017. On the numerical stability of surface–atmosphere coupling in weather and climate models. Geosci. model dev. 10(2), 977–989.

Burchard, H., Petersen, O., Rippeth, T.P., 1998. Comparing the performance of the mellor-yamada and the k-ϵ two-equation turbulence models. J. Geophys. Res. 103(C5), 10543–10554.

Duhaut, T., 2009. Notes sur les schémas de turbulence à deux équations. Technical Report. PREVIMER report.

McWilliams, J.C., Huckle, E., Shchepetkin, A.F., 2009. Buoyancy effects in a stratified ekman layer. J. Phys. Oceanogr. 39(10), 2581–2599.

Shchepetkin, A.F., 2005. If-less KPP. ROMS/TOMS Workshop: Adjoint Modeling and Applications, La.

Jolla, CA, October 24-28 Available online at http://www.myroms.org/Workshops/ROMS2005/Nonlinear/AlexanderShchepetkin.pdf.

Shchepetkin, A.F., 2009. Recent developments of ROMS at UCLA. Physical Oceanography Review Symposium Chicago, IL, June 7-13 Available online at http://people.atmos.ucla.edu/alex/ROMS/Chicago2009Talk.pdf.

Umlauf, L., Burchard, H., 2003. A generic length-scale equation for geophysical turbulence models. Journal of Marine Research 61(2), 235–265.

Umlauf, L., Burchard, H., 2005. Second-order turbulence closure models for geophysical boundary layers. a review of recent work. Continental Shelf Research 25(7), 795 – 827. Recent Developments in Physical Oceanographic Modelling: Part II.

A Implementation of GLS in Croco

The objective of this section is to describe the current implementation of a Generic Length Scale (GLS) turbulence scheme in Croco to determine K_m and K_s in (1.1). First of all, as usually done in most implementations, the assumption of an horizontally homogeneous flow is made and vertical advection is neglected. Following Umlauf and Burchard (2003), the equations satisfied by the two prognostic variables e (the kinetic energy) and ψ (the generic length scale) are

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

where the β_j (j=1,3) are constants to be defined, P represents the TKE production by vertical shear $P = K_m\left[(\partial_z u)^2 + (\partial_z v)^2\right]$ and B the TKE destruction by stratification $B = -K_s N^2$ (with N^2 the local Brunt-Vaisala frequency). The dissipation rate ε is related to the generic length scale ψ following

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

with l a mixing length and c_μ^0 a constant (whose value is between 0.526 and 0.555) to be defined. Depending on the parameter values for the triplet (m, n, p) the GLS scheme will either correspond to a $k-\varepsilon$, a $k-\omega$ or the so-called generic (Umlauf and Burchard, 2003) turbulence scheme [1] .

Since the equations for e and ψ bear lots of similarities, to avoid excessive code duplication, a unique equation is solved for a quantity $\mathcal{T}_i$ encompassing e (when $i = i_{\mathrm{tke}}$) and ψ (when $i = i_{\mathrm{gls}}$, $i_{\mathrm{gls}} = i_{\mathrm{tke}} + 1$) such that

$$\partial_t \mathcal{T}_i = \partial_z(K_{\mathcal{T}_i} \partial_z \mathcal{T}_i) + (c_i^1 P + c_i^{3,\pm} B - c_i^2 \varepsilon), \qquad K_{\mathcal{T}_i} = K_m/\mathrm{Sc}_{\mathcal{T}_i}$$

where

$$\mathrm{Sc}_{\mathcal{T}_{i_{\mathrm{tke}}}} = \mathrm{Sc}_e, \qquad \mathrm{Sc}_{\mathcal{T}_{i_{\mathrm{gls}}}} = \mathrm{Sc}_\psi$$

and

$$\begin{aligned} c_i^1 &= (i_{\mathrm{gls}} - i) + (i - i_{\mathrm{tke}})\beta_1 e^{-1}\psi \\ c_i^2 &= (i_{\mathrm{gls}} - i) + (i - i_{\mathrm{tke}})\beta_2 e^{-1}\psi \\ c_i^{3,\pm} &= (i_{\mathrm{gls}} - i) + (i - i_{\mathrm{tke}})\beta_3^{\pm} e^{-1}\psi \end{aligned}$$

[1] to simplify the code and because this scheme do not generally outperform other schemes, the possibility to use the so-called k-kl scheme is not implemented in Croco.

GLS model	m	n	p	β_1	β_2	β_3^-	β_3^+	Sc_e	Sc_ψ
k-ω	0.5	-1	-1	0.555	0.833	-0.6	1	0.5	0.5
k-ε	1.5	-1	3	1.44	1.92	-0.4	1	1	0.7692
Gen	1	-0.67	0	1	1.22	0.05	1	1.25	0.9345

Table A.1
Parameter values corresponding to each particular GLS model.

In practice this explains why in the code the two prognostic quantities e and ψ are stored in a single array $\mathrm{trb(i,j,k,ntime,ngls)}$ avec $\mathrm{ngls}=2$, $i_{\mathrm{tke}}=1$ and $i_{\mathrm{gls}}=2$. Once the quantities e and ψ (hence ε) are know, the turbulent viscosity/diffusivity are given by

$$K_m = c_\mu \left(\frac{e^2}{\varepsilon}\right) = \frac{c_\mu}{(c_\mu^0)^3}(l\sqrt{e}), \qquad K_s = c_\mu^{'} \left(\frac{e^2}{\varepsilon}\right) = \frac{c_\mu^{'}}{(c_\mu^0)^3}(l\sqrt{e}).$$

where c_μ and $c_\mu^{'}$ are determined through so-called stability functions (see below).

Choice of parameter values and stability functions

A particular GLS occurence is defined by the following parameters :

- The exponents (m,n,p) in the definition of ε
- The Schmidt numbers Sc_e and Sc_ψ
- The coefficients β_j (j=1,3)
- The constant c_μ^0
- The stability functions which are generally function of

$$\alpha_M = \left(\frac{e}{\varepsilon}\right)^2 \left[(\partial_z u)^2 + (\partial_z v)^2\right], \qquad \alpha_N = \left(\frac{e}{\varepsilon}\right)^2 N^2 \tag{A.1}$$

Where (m,n,p), Sc_e, Sc_ψ, β_j $(j=1,3)$ are tied to a particular choice of GLS scheme (see Tab. A.1) while c_μ^0, c_μ and $c_\mu^{'}$ are tied to a particular choice of stability function. The formulation of numerous stability functions can be reconciled when written using the generic form

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

where a given choice of stability function will define the parameter values for n_i, d_j, and $n_k^{'}$. In Croco, 7 options are available, these are referred to CANUTO-A, CANUTO-B, Gibson & Launder (1978), Mellor & Yamada (1982), Kantha & Clayson (1994), Luyten (1996), Cheng (2002).

The quantities α_N and α_M in the formulation of c_μ and $c_\mu^{'}$ must satisfy some constraints to guarantee the regularity of numerical solutions. In Croco, the following steps are done

(1) Apply the Galperin (1988) limitation i.e. $l \leq l_{\mathrm{lim}} = \beta_{\mathrm{galp}}\sqrt{2e/N^2}$ on ψ with $\beta_{\mathrm{galp}} = 0.53$. The first step is to use this mixing length l_{lim} to compute $\psi_{\mathrm{min}} = (c_\mu^0)^p e^m (l_{\mathrm{lim}})^n$ and to correct ψ to satisfy the constraint

$$\psi = \max(\psi, \psi_{\mathrm{min}})$$

here the max function is used since the exponent n is negative whatever the GLS scheme.

(2) Compute the dissipation rate $\varepsilon = (c_\mu^0)^{3+p/n} e^{3/2+m/n} \psi^{-1/n}$ and correct it

$$\varepsilon = \max\left(\varepsilon, \varepsilon_{\min}\right), \qquad \varepsilon_{\min} = 10^{-12}\ \mathrm{m}^2\ \mathrm{s}^{-3}$$

(3) Compute α_N and α_M using (A.1), and apply "stability and realisability" constraints following Umlauf and Burchard (2003) (Sec. 4). A first constraint applies on α_N to ensure that $-\partial_{\alpha_N}(c_\mu^{'}/\alpha_N) > 0$ to prevent the occurence of oscillations in $c_\mu^{'}$. This translates into the following limiter

$$\alpha_N^{\min} = \frac{-(d_1 + n_0^{'}) + \sqrt{(d_1 + n_0^{'})^2 - 4d_0(d_4 + n_1^{'})}}{2(d_4 + n_1^{'})}, \qquad \alpha_N = \min\left(\max(0.73\alpha_N^{\min}), 10^{10}\right)$$

where the coefficient 0.73 is used to ensure the so-called realisability and has been empirically computed thanks to Table 3 in Umlauf and Burchard (2003) in order to satisfy their constraint (48). Then an upper limit is applied on α_M to ensure that $\partial_{\alpha_M}(c_\mu\sqrt{\alpha_M}) \geq 0$ which is also a prerequisite for stability reasons

$$\alpha_M^{\max} = \frac{d_0 n_0 + (d_0 n_1 + d_1 n_0)\alpha_N + (d_1 n_1 + d_4 n_0)\alpha_N^2 + d_4 n_1 \alpha_N^3}{d_2 n_0 + (d_2 n_1 + d_3 n_0)\alpha_N + (d_3 n_1)\alpha_N^2}, \qquad \alpha_M = \min\left(\alpha_M, \alpha_M^{\max}\right)$$

Once those quantities are computed, stability functions are evaluated as well as the turbulent viscosity/diffusivity.

Surface and bottom boundary conditions

In current version of Croco, both e and ψ are formulated with Neumann boundary conditions at the top and at the bottom. However the nature of those boundary conditions also requires the determination of bottom and surface values for e and ψ.

- For turbulent kinetic energy, the "diagnostic" surface and bottom values are given by

$$e_{\mathrm{sfc}} = (u_\star^s / c_\mu^0)^2, \qquad e_{\mathrm{bot}} = (u_\star^b / c_\mu^0)^2$$

and simple homogeneous Neumann boundary conditions are applied

$$K_e \partial_z e|_{\mathrm{sfc}} = 0, \qquad K_e \partial_z e|_{\mathrm{bot}} = 0$$

In practice, due to the placement of e and ψ on the computational grid, the Neumann boundary condition is not applied strictly at the surface (resp. at the bottom) but at $z = z_N$ (resp. $z = z_1$) whereas the surface (resp. bottom) is located at $z = z_{N+1/2}$ (resp. $z = z_{1/2}$) with N the number of vertical levels (i.e. the number of cells in the vertical).

- For the generic length scale, a roughness is defined as

$$z_{0,s} = \max\left\{10^{-2}\ \mathrm{m}, \frac{C_{\mathrm{ch}}}{g}(u_\star^s)^2\right\}, \qquad C_{\mathrm{ch}} = 1400$$

at the surface and

$$z_{0,b} = \max\left\{10^{-4}\ \mathrm{m}, \mathrm{Zob}\right\}$$

at the bottom with Zob a user defined roughness length (usually $\mathrm{Zob} = 10^{-2}$ m). Again, the boundary conditions are applied at the center of the shallowest and deepest grid cells and not at their interfaces

which means that the relevant length scales are

$$L_{\rm sfc} = \kappa\left(\frac{\Delta z_N}{2} + z_{0,s}\right), \qquad L_{\rm bot} = \kappa\left(\frac{\Delta z_1}{2} + z_{0,b}\right)$$

with κ the von Karman constant. Moreover TKE values are interpolated at $z = z_N$ and $z = z_1$

$$\widetilde{e}_{\rm sfc} = \frac{1}{2}\left(e_{\rm sfc} + e_{\rm N-1/2}\right), \qquad \widetilde{e}_{\rm bot} = \frac{1}{2}\left(e_{\rm bot} + e_{3/2}\right)$$

where $e_{\rm sfc}$ and $e_{\rm bot}$ are the diagnostic values given above. The "diagnostic" surface and bottom values for ψ are thus given by

$$\psi_{\rm sfc} = (c_\mu^0)^p (L_{\rm sfc})^n (\widetilde{e}_{\rm sfc})^m, \qquad \psi_{\rm bot} = (c_\mu^0)^p (L_{\rm bot})^n (\widetilde{e}_{\rm bot})^m$$

Then the surface and bottom flux are defined as

$$\mathcal{F}_\psi^{\rm sfc} = K_\psi \partial_z \psi|_{\rm sfc} = -n(c_\mu^0)^{p+1} \frac{\kappa}{{\rm Sc}_\psi} (\widetilde{e}_{\rm sfc})^{m+1/2} (L_{\rm sfc})^n$$

$$\mathcal{F}_\psi^{\rm bot} = K_\psi \partial_z \psi|_{\rm bot} = -n(c_\mu^0)^{p+1} \frac{\kappa}{{\rm Sc}_\psi} (\widetilde{e}_{\rm bot})^{m+1/2} (L_{\rm bot})^n$$

which correspond to the Neumann boundary conditions applied in the code.