An energy fixer for CLUBB in the E3SM climate model

Vince Larson, Brian Griffin

Outline

1. What is the CLUBB turbulence parameterization?
2. Energy conservation in E3SM: RESTOM, RESSURF, and check_energy_chng
3. Why does CLUBB need an energy fixer?
4. Our attempts to improve CLUBB’s energy fixer

CLUBB is a parameterization of turbulence (and clouds)

CLUBB is a column-wise parameterization that transports subgrid-scale heat content, moisture, and momentum in the vertical.

CLUBB is used in E3SM and CAM.

CLUBB is written in terms of a z (height) coordinate, unlike E3SM

CLUBB uses the anelastic approximation.

CLUBB doesn’t change pressure of the height levels due to heating of air or evaporation of liquid; instead, that is left to the host model.

CLUBB's heat content variable is liquid water potential temperature, $theta_l$:

$theta_l = theta - (L/c_p)\ (Exner)^{-1}\ r_c$

where *theta* is potential temperature and r_c is cloud liquid water.

(The version of CLUBB in E3SM doesn't see ice.)

$theta_l$ is different than E3SM's conserved heat variable (moist static energy).

CLUBB’s equation for heat transport conserves *theta*$_l$ in the vertical

CLUBB vertically advects heat, moisture, and momentum. E.g., CLUBB delivers the right-hand side of

$$\frac{\partial \overline{\theta_l}}{\partial t} = -\frac{1}{\rho}\frac{\partial \rho \overline{w'\theta_l'}}{\partial z}$$

CLUBB delivers only this flux term, not net sources or sinks. Those are handled by, e.g., the microphysics or radiation.

CLUBB outputs tendencies based on fluxes

It would be natural to deliver the fluxes directly to the host model, although in E3SM tendencies are delivered.

Regardless, CLUBB's fluxes are of the wrong heat variable.

CLUBB's fluxes (and some other higher-order moments) are prognosed:

$$\mathrm{Means:} \quad \frac{\partial \overline{u}}{\partial t} = \ldots \quad \frac{\partial \overline{v}}{\partial t} = \ldots \quad \frac{\partial \overline{r_t}}{\partial t} = \ldots \quad \frac{\partial \overline{\theta_l}}{\partial t} = \ldots$$

$$\mathrm{2nd-order:} \quad \frac{\partial \overline{w'r_t'}}{\partial t} = \ldots \quad \frac{\partial \overline{w'\theta_l'}}{\partial t} = \ldots \quad \frac{\partial \overline{w'^2}}{\partial t} = \ldots$$

$$\frac{\partial \overline{r_t'^2}}{\partial t} = \ldots \quad \frac{\partial \overline{\theta_l'^2}}{\partial t} = \ldots \quad \frac{\partial \overline{r_t'\theta_l'}}{\partial t} = \ldots$$

$$\mathrm{3rd-order:} \quad \frac{\partial \overline{w'^3}}{\partial t} = \ldots$$

$w =$ vertical velocity

$r_t =$ total water mixing ratio

$\theta_l =$ liquid water potential temperature

However, CLUBB’s prognostic equations sit on top of a diagnostic, Monin-Obukhov surface layer

Diagnostic schemes tend to have two drawbacks:

- they don’t have memory; and
- they have poor time step convergence at small time steps.

(Does this cause problems?)

To increase the time step, CLUBB uses semi-implicit (backward Euler) time stepping

The vertical discretization is either 1st-order upwind or 2nd-order centered differences. A higher-order method would be unwieldy and slow.

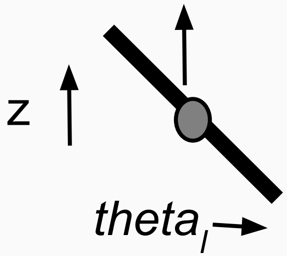
Hence, the prospects for higher-order discretization are poor.

CLUBB includes turbulent potential energy through the prognosis of temperature variance, d/dt($theta_l'^2$)

The turbulent potential energy, $theta_l'^2/N^2$, as discussed by Zilitinkevich et al. (2008), is useful for conceptualization of stably stratified layers, but it is not convenient for calculations of neutrally stratified layers, where $N^2 = 0$.

Where does the physics of upgradient transport reside?

$$\frac{\partial \overline{w'\theta_l'}}{\partial t} = \underbrace{-\overline{w'^2}\frac{\partial \overline{\theta_l}}{\partial z}}_{\text{Turb Prod}} \underbrace{+(1-C_5)\frac{g}{\theta_{vs}}\overline{\theta_l'\theta_v'}}_{\text{Buoy Prod}} \underbrace{-\frac{C_4}{\tau}\overline{w'\theta_l'}}_{\text{Pressure}} \underbrace{-\frac{\partial \overline{w'^2\theta_l'}}{\partial z}}_{\text{Turb Transport}}$$



pressure damps heat flux with a time scale tau

turb transport is a flux of flux:

$w'^{2}\mathit{theta}_l' = w'(w'\mathit{theta}_l')$

. . . in the buoyancy production and turbulent transport (flux-of-flux) terms:

To see this, drop the time tendency term and re-arrange:

$$\overline{w'\theta_l'} = \underbrace{-\frac{\tau}{C_4}\overline{w'^2}\frac{\partial\overline{\theta_l}}{\partial z}}_{\text{Eddy Diff}} \underbrace{+\frac{\tau}{C_4}(1-C_5)\frac{g}{\theta_{vs}}\overline{\theta_l'\theta_v'}}_{\text{Buoy Prod}} \underbrace{-\frac{\tau}{C_4}\frac{\partial\overline{w'^2\theta_l'}}{\partial z}}_{\text{Turb Transport}}$$

The turbulent production term leads to downgradient diffusion, with diffusivity $K = tau <w'^2>/C_4$. There is widespread disagreement about how to handle the upgradient terms.

For future reference: I note that CLUBB also updates momentum

$$\frac{\partial \overline{u}}{\partial t} = -\frac{1}{\rho}\frac{\partial \rho \overline{u'w'}}{\partial z}$$

Surface drag drains momentum from the column.

But diffusion of momentum does not generate dissipative heating in CLUBB.

Outline

1. What is the CLUBB turbulence parameterization?
2. Energy conservation in E3SM: RESTOM, RESSURF, and check_energy_chng
3. Why does CLUBB need an energy fixer?
4. Our attempts to improve CLUBB's energy fixer

Energy conservation in climate models such as E3SM

What interests us here is conservation of energy of a physics parameterization in an individual grid column (not global conservation).

A heat-content variable is conserved (*S*) if we account for downward top-of-model flux (*RESTOM*) and surface flux (*RESSURF*)

A conservation equation may be written as (ignoring horizontal advection):

$dS/dt = g\, d/dp(w'S') + Source$

This may be vertically integrated to yield:

$d <S\, dp/g>/dt = RESTOM - RESSURF + < Source\, dp/g >$

After long times, if there is no source, and if the model reaches an equilibrium, then *RESTOM* = *RESSURF*.

Troubleshooting: To test whether a particular parameterization is conservative, use subroutine *check_energy_chng*

check_energy_chng calculates, for a given subroutine, the quantity

$$\begin{aligned} &< S\ dp/g >_{after} - < S\ dp/g >_{before} \\ &\qquad + \quad (- RESTOM + RESSURF) * dt \end{aligned}$$

This quantity should equal zero to within round-off or so, if the parameterization is conservative.

Outline

1. What is the CLUBB turbulence parameterization?
2. Energy conservation in E3SM: RESTOM, RESSURF, and check_energy_chng
3. Why does CLUBB need an energy fixer?
4. Our attempts to improve CLUBB's energy fixer

When implemented in E3SM, CLUBB needs a column-local energy fixer because . . .

. . . CLUBB's heat content variable (*theta*$_l$) is different than E3SM's (moist static energy)

Use of a different energy variable in CLUBB leads to a different weighting in the vertical integral of energy (thanks to Chris Golaz for the derivation)

In CLUBB, the vertical integral of *theta*$_l$ is constant (neglecting surface fluxes for simplicity of discussion):

$$\int c_p \Delta\theta_l \rho \, dz = 0$$

Substituting in *theta*$_l$ = *theta - (L/c*$_p$*) (Exner)*$^{-1}$ *r*$_c$ and

theta = T (Exner)$^{-1}$, we find:

$$\int \Pi^{-1}(c_p \Delta T - L\Delta r_c)\rho \, dz = 0$$

E3SM uses moist static energy rather than theta_l

In E3SM, the vertical integral of moist static energy,

$s_l = c_p T + gz - L\, r_c$,

is constant:

$$\int \Delta s_l \rho \, dz = 0$$

Substituting the definition of moist static energy, we find:

$$\int (c_p \Delta T - L \Delta r_c) \rho \, dz = 0$$

CLUBB’s energy integral includes Exner, and E3SM’s does not

CLUBB’s conservation integral,

$$\int \Pi^{-1}(c_p \Delta T - L \Delta r_c)\rho\, dz = 0$$

is weighted by *(Exner)*$^{-1}$, whereas E3SM’s is not:

$$\int (c_p \Delta T - L \Delta r_c)\rho\, dz = 0$$

This is the reason for the non-conservation.

Outline

1. What is the CLUBB turbulence parameterization?
2. Energy conservation in E3SM: RESTOM, RESSURF, and check_energy_chng
3. Why does CLUBB need an energy fixer?
4. Our attempts to improve CLUBB’s energy fixer

Status quo in E3SMv1: CLUBB's energy fixer is based on *check_energy_chng*

Code is copied from *check_energy_chng* that computes the vertical integral of TKE + moist static energy.

The vertical integral is computed before and after CLUBB.

If there is an imbalance, a temperature increment is applied uniformly with height.

This results in fractional errors of ~1e-4 per time step.

Notice that the derivation assumes that there is no change in the altitude of the pressure levels. Could this be a problem?

On physical grounds, we expect that deposition of heat in a layer will push apart the height levels:

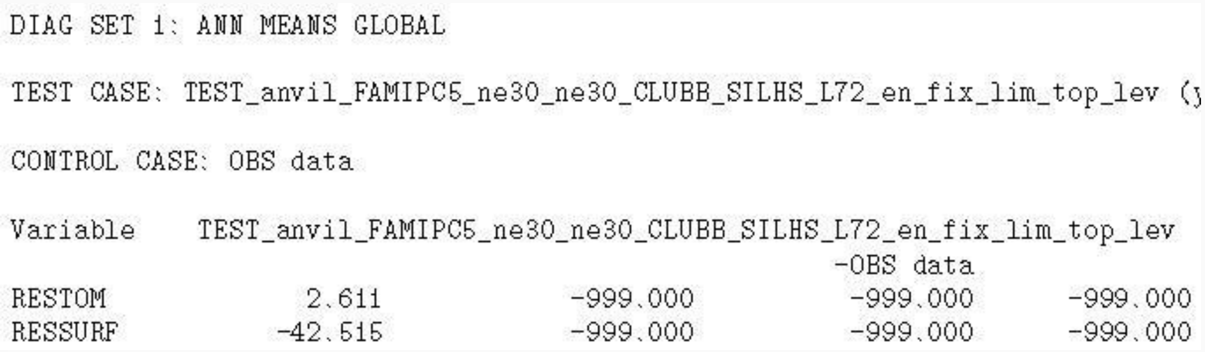
$$dp/dz = - \rho g$$

$$dz = - (R_d T / g) dp/p$$

So Brian Griffin included $g\, dz$ in CLUBB’s energy fixer.

If we include geopotential height, *gdz*, in the energy fixer, then

RESTOM
 -RESSURF
= 44 W/m^2 (!):

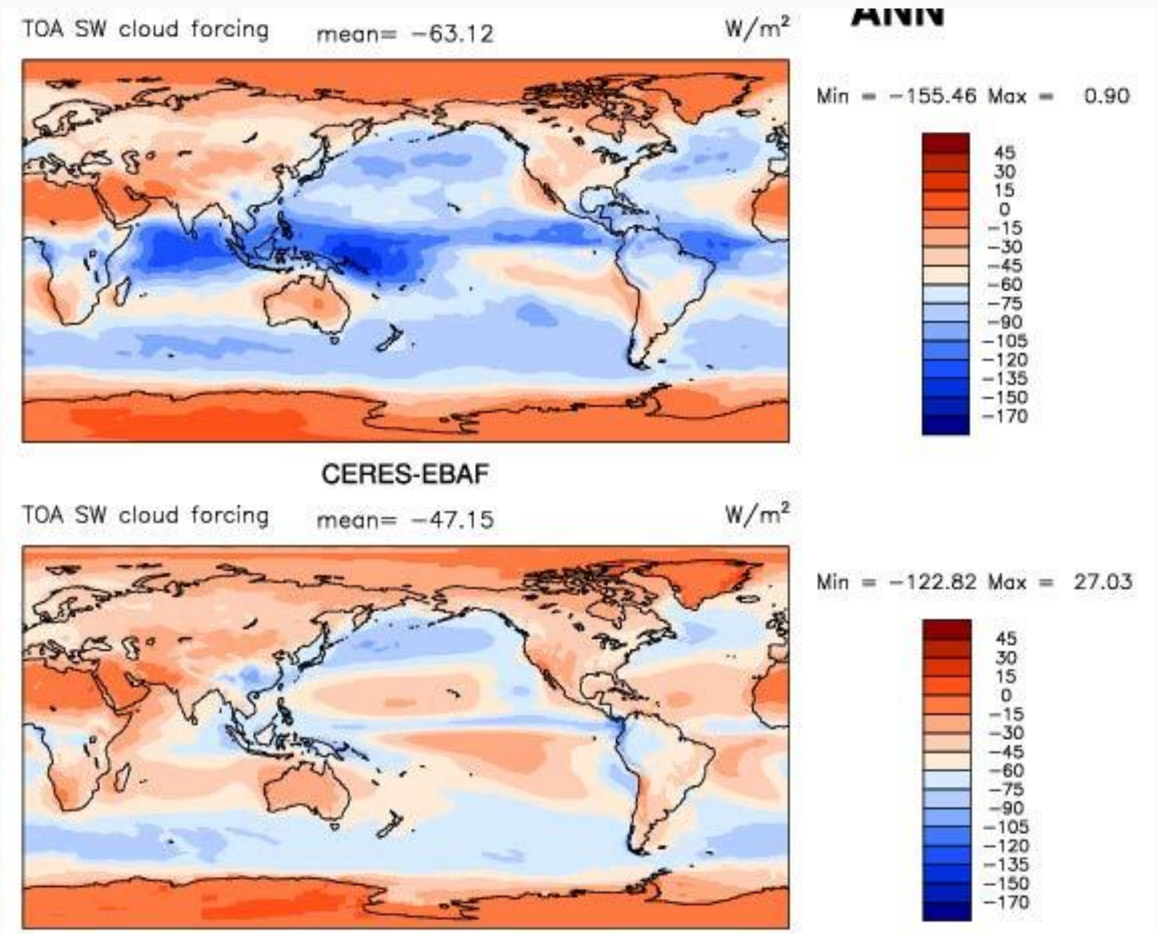


```
DIAG SET 1: ANN MEANS GLOBAL

TEST CASE: TEST_anvil_FAMIPC5_ne30_ne30_CLUBB_SILHS_L72_en_fix_lim_top_lev (

CONTROL CASE: OBS data

Variable    TEST_anvil_FAMIPC5_ne30_ne30_CLUBB_SILHS_L72_en_fix_lim_top_lev
                                                        -OBS data
RESTOM             2.611          -999.000          -999.000        -999.000
RESSURF          -42.515          -999.000          -999.000        -999.000
```



The problem is that the *gz* term should not be included in E3SM’s energy variable because of E3SM’s pressure coordinate:

Williamson et al. (2015) revised the energy variable:

E3SM’s old energy variable:

$d/dt\ (K.\ E. + c_p T + gz) = 0$

E3SM’s corrected energy variable:

$d/dt\ (K.\ E. + c_p T) = 0$

Use of Williamson’s energy variable requires us to modify *check_energy_chng*

```
          se(i) = se(i) + state%s(i,k )*state%pdel(i,k)/gravit
!!! cam6  se(i) = se(i) + state%t(i,k)*cpairv_loc(i,k,lchnk)*state%pdel(i,k)/gravit
```

The commented-out line calculates the correct energy.

A further problem for CLUBB: Surface drag is not included in *check_energy_chng*

$$\frac{\partial \overline{u}}{\partial t} = -\frac{1}{\rho}\frac{\partial \rho \overline{u'w'}}{\partial z}$$

check_energy_chng triggers an error whenever <rho u^2> changes during the call to CLUBB.

Of course, *<rho u^2>* is changed by CLUBB because of surface drag.

Brian Griffin's temporary workaround: Exclude kinetic energy from *check_energy_chng*

This is a reasonable thing to do for CLUBB, because CLUBB does not produce dissipative heating. Therefore, CLUBB's moist static energy should be conserved independently of kinetic energy.

When Brian Griffin uses Williamson's energy, minus the kinetic energy term, then he finds that CLUBB+fixer is conservative to within a fractional error of 1e-10 per time step.

I have replaced CLUBB's original, malfunctioning energy fixer with a

`new energy fixer that is William F. Buckley Jr. conservative!`

bmg929/atm/clubb_silhs_v2

bmg929 committed on Jul 24 `1 parent 0fc606d` `commit 5262bfdfde7e974ce552bb80a7f67b61a48f6cec`

Possible next steps

1. check_energy_chng itself needs to be verified
2. check_energy_chng ought to be used in routine tests

Thanks!

Extra slides

Conservation of water due to microphysics

Suppose the model's time step is too long. Autoconversion depletes cloud water until the cloud water is highly negative at the end of the time step:

rcloud_n + drcloud/dt * dt << 0

A huge amount of rain is generated by autoconversion. Then sedimentation moves all the generated rain downward.

Problem: Because of fallout, there may not be enough water (rain) mass to fill the hole

In principle, this problem can be avoided by using sequential splitting of local microphysics tendencies and sedimentation, and clipping after the local tendencies are computed.

We'll be talking about the case where the model doesn't clip between local processes and sedimentation

In this case, we can fill the holes in a conservative manner by treating the local tendencies and the sedimentation tendencies separately.

In this example, autoconversion is the local process.

Step 1: Some of the local process rates may be negative and some may be positive

To fill the hole, we need to reduce the magnitude of drcloud/dt. But we need to do so in a way that conserves total water. We assume that the sum of the local tendencies is zero:

drcloud/dt + drrain/dt = 0

I.e. autoconversion creates as much rain as cloud is lost.

Step 2: To conserve total water, we may reduce (“clip”) the magnitude of the local positive tendencies (rain) proportionately

drcloud/dt + drrain/dt = 0

The excess rain created by excessive autoconversion is clipped back.

(We repeat this step for all species that generate negative values (“holes”).)

Step 3: Now there is a new problem: sedimentation may overdeplete rain, because the autoconversion tendency was clipped back

So now we fill the hole in rain by vertical hole filling.

We draw rain mass from the altitudes below the level with the hole, plus we pull back rain that fell into the ocean if needed.

We can guarantee that all holes will be filled.