A fast and complete convection scheme for ocean models

Stefan Rahmstorf

Institut für Meereskunde
Kiel
Germany

published in

Ocean Modelling Nr. 101

(1993)

Imagine having three half-filled glasses of wine lined up in front of you. On the left a German Riesling, in the middle a French Burgundy and on the right a Chardonnay from New Zealand. Imagine further that you're not much of a connoisseur, so you want to mix the three together to a refreshing drink, with exactly the same mixture in each glass. The trouble is, you can only mix the contents of *two* adjacent glasses at a time. So you start off by mixing the Riesling with the Burgundy, then you mix this mixture with the Chardonnay, then... How often do you need to repeat this process until you get an identical mix in all glasses?

Incidentally, putting this question to a friend is a good test to see whether she (or he) is a mathematician or a physicist. A mathematician would answer "an infinite number of times", while a physicist would be well aware that there is only a finite number of molecules involved, so you can get your perfect drink with a finite mixing effort (only you would have no way to tell whether you've got it or not).

In any case, the number of times you need to mix is very large, and this is the problem of the standard convection scheme of the GFDL ocean model (Cox 1984), which mixes two adjacent levels of the water column if they are statically unstable. The model includes the option to repeat this mixing process a number of times at each time step, as an iteration process towards complete removal of static instabilities. The minimum number of iterations needed to mix *some* of the information from layer 1 down to layer n is n-1.

To avoid this problem, one needs to relax the condition that only two levels may be mixed at a time. To achieve complete mixing, a convection scheme is required that can mix the whole unstable part of the water column in one go. I have been using such a scheme back in 1983 in a one-dimensional mixing model for the Irish Sea, and I'm sure many other people have been using similar ones. Marotzke (1991) introduced such a scheme into the GFDL ocean model. It appears that it hasn't been taken up as enthusiastically as it might have been, and an implicit convection scheme (which increases the vertical diffusivity at unstable parts of the water column) has been preferred because of lower computational cost (e.g. Weaver et al, 1993). However, it is not difficult to set up a complete convection scheme which uses less computer time than the implicit scheme.

The standard scheme

Since the GFDL model works the grid row by row, we'll only discuss how one grid-row is treated. Here's how:

(1) Compute the densities for all grid cells in the row. Two adjacent levels are always referenced to the same pressure in order to get the static stability of this pair of levels.
(2) Mix all unstable pairs.
(3) Since we have now only compared and mixed "even" pairs (i.e. levels 1 & 2; levels 3 & 4; etc), repeat steps (1) and (2) for "odd" pairs (i.e. levels 2 & 3; levels 4 & 5; etc).
(4) Repeat steps (1)-(3) a predetermined number of times.

There is a couple of problems here. We've already said that strictly speaking this never leads to complete mixing of an unstable water column. So the process is repeated several times at each time step to approximate complete mixing. But each time *all* grid cells are checked for instabilities again, even those we already found to be stable. Each density calculation requires evaluation of a third order polynomial in T and S. This is where the cpu time is eaten up.

Marotzke's scheme

This scheme works as follows:

(1) Same as step (1) above, except that the stability of *all* pairs of grid cells is checked, odd and even pairs (so that the density of interior levels is computed twice, for two different reference pressures).

(2) Don't mix yet: just mark all unstable pairs and find continuous regions of the water column which are unstable (neutral stability is treated as unstable).

(3) Mix the unstable regions.

(4) If there was instability in any column, repeat steps (1) to (3). Those columns which were completely stable in the previous round are not dealt with again in (2) and (3), but the densities are still recomputed for the entire grid row. Repeat until no more instabilities are found.

So Marotzke relaxed the condition that only two levels are mixed at a time, and complete mixing will be achieved with at most *k-1* passes through the water column, if *k* is the number of model levels. However, if only one grid point of a row requires *n* iterations, the densities for the entire grid row will be recomputed *n* times, so it still doesn't look too good in terms of cpu efficiency.

The fast way

(1) Compute all densities like in (1) of Marotzke.

(2) Compare all density pairs to find instabilities.

From here on, deal column by column with those grid points where an instability was found, performing the following steps:

(3) Mix the uppermost unstable pair.

(4) Check the next level below. If it is less dense than the mixture, mix all three. Continue incorporating more levels in this way, until a statically stable level is reached.

(5) Then check the level *above* the newly mixed part of the water column, to see whether this has become unstable now. If so, include it in the mixed part and go back to (3). If not, search for more unstable regions below the one we just mixed, by working your way down the water column comparing pairs of levels; if you find another unstable pair, go to (3).

Note that levels which have been mixed are from then on treated as a unit. This scheme has a slightly more complicated logical structure; it needs a few more integer variables and *if* statements to keep track of which part of the water column we have already dealt with. The advantage is that we only recompute the densities of those levels we need; levels which are not affected by the convection process are only checked once. The scheme includes diagnostics which allow to plot the convection depth at each grid point.

Discussion

Perhaps these schemes are best discussed with an example. Imagine a model with five levels. At one grid point levels 2 & 3 and levels 3 & 4 are statically unstable. The standard scheme will, at the first pass, mix 3 & 4 and then 2 & 3. It will repeat this *ncon* times. Marotzke's scheme will mark the unstable pairs and then mix 2-4 in one go. It will then return to this column for a second pass and check all levels once more. My scheme will mix 2 & 3; then compare the densities of 3 & 4 and (if unstable) mix 2-4 like Marotzke's scheme. It will then recompute the density of level 4, compare levels 4 & 5 and mix 2-5 if unstable. Finally it will compare 1 & 2 again, since the density of 2 has changed in the mixing process, so level 1 might have become statically unstable. Only the density of 2 is recalculated for this.

Note that Marotzke's scheme handles the initial mixing of levels 2-4 more efficiently. Probably my scheme could be made slightly faster still by including the "marking" feature from Marotzke's scheme (the schemes were developed independently). However, in the typical convection situation only levels 1 & 2 are initially unstable, due to surface cooling. In this situation marking doesn't help. My scheme saves time by "remembering" which parts of the water column we already know to be stable, and rechecking only those levels necessary.

There is a subtlety that should be mentioned: due to the non-linear equation of state the task of removing all static instability from the water column may not have a unique solution. In the example above, mixing 2 & 3 could yield a mixture with a lower density than level 4, in spite of 3 being denser than 4, and 2 being denser than 3 originally. In this case, my scheme would only mix 2 & 3, while Marotzke's scheme would still mix 2-4. So both schemes are not strictly equivalent, though for all practical purposes they almost certainly are.

I performed some test runs with the GFDL modular ocean model (MOM) in a two-basin configuration (the same as used by Marotzke and Willebrand 1991). The model has ca. 1000 horizontal grid points and 15 levels, and was integrated for 1 year (time step 1.5 h) on a Cray YMP. Three different model states were used: (A) a state with almost no static instability, achieved by strong uniform surface heating; (B) a state with convection occurring at about 15% of all grid points; (C) a state with convection at 30% of all grid points. The latter two were near equilibrium, with permanent convection. I compared the overall cpu time consumed by these runs with different convection schemes. The standard scheme was tried for three different numbers of iterations *ncon.* The results are summarised in the table; the overall cpu time is given relative to a run with no convection scheme.

Convection scheme	relative cpu time		
	A	B	C
No convection scheme	1	1	1
standard, *ncon=1*	1.13	1.13	1.13
standard, *ncon=7*	1.88	1.89	1.92
standard, *ncon=10*	2.25	2.27	2.32
implicit	1.52	1.52	1.52
complete	1.12	1.20	1.36

I was surprised to find that the few innocent-looking lines of model code that handle the convection consume a large percentage of the overall processing time. The numbers are probably an upper limit; a model with realistic topography and time-dependent forcing will use a bigger chunk of the cpu time for iterations in the relaxation routine for the stream function, so that the *relative* amount spent on convection will be lower. In my test runs, the standard scheme adds 13% cpu time per pass. My complete convection scheme used as much time as 1-3 iterations of the standard scheme, depending on the amount of convection. For zero convection it is as fast as one pass of the standard scheme, because it does the same job in this case. Additional cpu time is only used at those grid points where convection actually occurs. My scheme is considerably faster than the implicit scheme, especially for models where convection happens only at a few grid points, or only part of the time. I did not have Marotzke's scheme available for the test, but in his 1991 paper he mentions a comparison where the computation time with the implicit scheme was 60% of that with his scheme. This would give Marotzke's scheme a relative cpu time of about 2.5 in the table, with strong dependence on the amount of convective activity.

Surface heat fluxes looked identical in the runs with the implicit and complete schemes. The standard scheme showed significant deviations, however, in the surface flux as well as the convective heat flux at different depths. This is not surprising, since the rate at which heat is brought up by convection will be reduced if mixing is incomplete. The runs with *ncon=7* and *ncon=10* still differed noticeably from each other, and from the complete mixing case. It is possible that this could affect the deep circulation, which is driven by convective heat loss, but I didn't do long integrations to test this. The problem gets worse for longer time steps; with the standard scheme the rate of vertical mixing depends on the time step length. If acceleration techniques are used ("split time stepping", Bryan 1984), the final equilibrium could differ from one without acceleration due to this unwanted time-step dependence. Marotzke (1991) reports a case where the choice of convective scheme had a decisive influence on the deep circulation. The intention of this note is not to examine these problems any further; it is to provide an efficient alternative.

Conclusion

I have described a convection scheme which completely removes static instability from the water column in one pass, and which is much faster than the implicit scheme of the GFDL model. This scheme avoids possible problems resulting from the inclomplete mixing in the standard scheme, while only using as much computer time as 1-3 iterations of the standard scheme.

References

BRYAN K. (1984) Accelerating the convergence to equilibrium of ocean-climate models. *Journal of Physical Oceanography,* **14**, 666-673.

COX M. D. (1984) *GFDL Ocean Group Technical Report,* 1: *A primitive equation, 3-dimensional model of the ocean,* GFDL, Princeton University, Princeton, 143 pp.

MAROTZKE J. (1991) Influence of convective adjustment on the stability of the thermohaline circulation. *Journal of Physical Oceanography,* **21**, 903-907.

MAROTZKE J. and J. WILLEBRAND (1991) Multiple equilibria of the global thermohaline circulation. *Journal of Physical Oceanography,* **21**, 1372-1385.

WEAVER A. J., E. S. SARACHIK and J. MAROTZKE (1991) Freshwater flux forcing of decadal and interdecadal oceanic variability. *Nature,* **353**, 836-838.

WEAVER A. J., J. MAROTZKE, P. F. CUMMINS and E. S. SARACHIK (1993) Stability and variability of the thermohaline circulation. *Journal of Physical Oceanography,* **23**, 39-60.