On the Computational Efficiency of Ocean Models:

A High Performance Equation of State

Dov Kruger[1], Tal Ezer[2], Anne M. Pence[1] and Alan F. Blumberg[1]

[1]Davidson Laboratory, Civil, Environmental and Ocean Engineering
Stevens Institute of Technology, Hoboken, NJ 07030

[2]Program in Atmospheric and Oceanic Sciences, Princeton University
P.O.Box CN710, Sayre Hall, Princeton, NJ 08544-0710
Corresponding author email: ezer@splash.princeton.edu

To be submitted to:
Journal of Atmospheric and Oceanic Technology

December 20, 2004

Abstract

The numerical representation of the equation of state (EOS) in ocean models is reevaluated in terms of its computational efficiency. Ways to speedup the EOS as well as basic codes in ocean models are discussed and demonstrated. It is shown that even a simple rearrangement of a calculation order can significantly speedup the codes currently used in several community terrain-following ocean models. Using a new EOS with particular fit to data that takes into account the computational cost, can speedup the code even further. Various EOS codes were tested on different computer platforms, demonstrating the potential of speeding up codes by as much as 2-3 times on a supercomputer and up to 6-16 times on laptops and personal computers.

1. Introduction

The equation of state, used to relate density of sea water to salinity, temperature and pressure, has concerned oceanographers for a long time (e.g., Fofonoff, 1956). While earlier efforts have used simple linear fits to data, a more accurate but complex equation of state (EOS) became the standard (UNESCO, 1981). However, the formulation was very computationally intensive, so ocean modelers have attempted to reduce the cost with improved formulations that may trade off accuracy for speed. For example, early z-level models, whose vertical grid follows constant depth levels, have used a table of coefficients for each level (Bryan and Cox, 1972). Simplified approach of this nature can not be used in non-aligned vertical grids such as in terrain-following ("s" or "sigma" coordinates) models (e.g., the Estuarine, Coastal and Ocean Model, ECOM, Blumberg et al., 1993; the Princeton Ocean Model, POM, Blumberg and Mellor, 1987; the Regional Ocean Modeling System, ROMS, Shchepetkin and McWilliams, 2005), or in generalized coordinate ocean models (Ezer and Mellor, 2004). Note that early sigma models were used mostly for shallow coasts and estuaries, so the pressure dependency of the EOS was often neglected. The extension of sigma ocean model applications to deep-ocean and basin-scale problems (Ezer and Mellor, 1997; Haidvogel et al., 2000) require to include

the pressure effect and led, for example, to the EOS suggested by Mellor (1991), which simplified the pressure term in the UNESCO formulation.

There are two factors limiting computational fluid dynamics models, memory and computational time. In the case of ocean modeling, the memory size is usually less of a problem (except in very high resolution or global models), but time is a limiting factor (e.g., calculations of a real-time daily forecast should take much less than a day to be useful). Thus, a faster computation means the ability to have a larger domain, a finer grid, or the ability to perform more experiments. There are three basic approaches for speeding up codes. The first approach is by using parallel codes (e.g., the ROMS code) that take advantage of multiprocessor supercomputers. The second approach is by using numerical schemes that allow longer time steps between calculations (Ezer et al., 2002; Shchepetkin and McWilliams, 2005). The third approach, discussed here, is by optimizing the basic code structure so more calculations can be done per computational time, taking into account the fact that some operations are faster than others. This approach is synergistic with the other approaches. Parallel computing is more efficient when each parallelized unit is faster, and in particular shared memory architectures are much more efficient when the algorithms are efficient of memory bandwidth and are CPU-limited.

While large multiprocessor supercomputers are widely used in ocean modeling, at the same time, the speed and memory available on personal computers, PCs (workstations, laptops, etc.), have increased dramatically in recent years, allowing users to run complex ocean models on local PCs. However, surprisingly little attention has been given to improvement in the performance of the basic code itself when using a single processor. It was discovered, that common ocean models may not be optimized for PCs, which was the motivation behind this study. This deficiency in the computational efficiency occur for example in the formulations of the EOS used in three community terrain-following ocean models: ROMS (Haidvogel et al., 2000; Shchepetkin and McWilliams, 2005) which uses the Jackett and McDougall (1995) formulation, POM (Blumberg and Mellor, 1987) which uses the Mellor (1991) formulation, and ECOM (Blumberg et al., 1993) which uses a version of Fofonoff (1956) without the pressure terms. Over 3000 users world wide use these community models and can benefit from improved codes. The above formulations are also compared with a new polynomial fit. The EOS in the above models accounts for ~10% of the total calculations, so optimizing

the EOS alone would speed the entire model by less than 10%. The most computationally demanding part in POM for example is the turbulence scheme (Mellor and Yamada, 1982) which takes ~20% of the entire model calculations, but preliminary tests (not shown) indicate the potential for speeding up the turbulence scheme and other parts of ocean model codes using similar concepts as demonstrated here with the EOS example.

2. **The formulation of the equation of state**

The original UNESCO expression, without the pressure dependent terms can be written in the form,

$$\begin{aligned}\rho(S,\theta) &= c_{00} + c_{11}\theta + c_{12}\theta^2 + c_{13}\theta^3 + c_{14}\theta^4 + c_{15}\theta^5 \\ &+ (c_{21} + c_{22}\theta + c_{23}\theta^2 + c_{24}\theta^3 + c_{25}\theta^4)S \\ &+ (c_{31} + c_{32}\theta + c_{33}\theta^2)S^{3/2} + c_{41}S^2\end{aligned} \tag{1}$$

where ρ, S and θ are the density, salinity and potential temperature, respectively. This formulation was based on fitting the expression to a large sample of oceanic observations in order to find the optimal coefficients c_{ij}. Note however, that the original formula used the in-situ temperature, T, while ocean models need the potential temperature, θ, where $\rho(S,T) = \rho(S,\theta,p=0)$, and p is pressure. The EOS in the standard POM, is written the same way as (1) (straight forward format, but apparently costly, as demonstrated later). The original complex UNESCO pressure dependent term that is added to (1) was simplified by Mellor (1991), which reduces the computational cost by a factor of 3 compared with the full UNESCO formulation. Other alternative pressure terms are also been developed for the other schemes proposed below, but they will be evaluated in detail in a separate paper, so the pressure term is ignored for now. Various ways to make (1) more computationally efficient have been tested. First, we can rearrange (1) into Horner's form (Knuth, 1973), and turn the 3/2 power to a square root call, so it can be written as,

$$\begin{aligned}\rho(S,\theta) &= c_{00} + ((((c_{11}\theta + c_{12})\theta + c_{13})\theta + c_{14})\theta + c_{15})\theta + \\ &(+(c_{21} + c_{22}\theta + c_{23}\theta^2 + c_{24}\theta^3 + c_{25}\theta^4 + c_{41}S))S + (c_{31} + c_{32}\theta + c_{33}\theta^2)\sqrt{S}S\end{aligned} \tag{2}$$

With this form we can achieve an immediate speedup factor of up to ~3 on a PC (see detail results on various platforms later). This is due to the expense of exponentiation, and the reduction in the number of multiplications. This optimized form of the UNESCO EOS is proposed to be implemented in ECOM. Note also that most compilers may not optimize writing to an array, then reading from it and writing to it again. For example, the code may calculate the density, ρ(x,y,z)=rho(i,j,k), where i,j,k are the model grid indexes and then normalize the results by a reference value, rhoref, and multiply by a land/ocean mask, fsm(i,j). However, even simply putting the expression in a single statement and rolling the rhoref constant into the coefficients can further reduce the cost. The remaining code in (2) is nearly optimal, but it still contains a square root (relatively slow operation), and includes many terms. One may ask a fundamental question: why was this fit in (1) chosen in the first place? Is the odd power important for some reason? When we looked at the original paper, entered the data, and did our own fit, we found no compelling reason to fit to $S^{3/2}$. The error between repeated non-dimensional density measurements is on the order of 10^{-3} (measurements are in the form of non-dimensional difference from the standard density of seawater). When comparing the computed density using the new fit with the data, the error is of the same order of magnitude, also similar to the error between UNESCO and the data. Therefore the error of the fit is within the measurement error. The proposed new fit, of the form

$$\begin{aligned}\rho(S,\theta) = c_0 + ((((c_1\theta + c_2)\theta + c_3)\theta + c_4)\theta + c_5)\theta + (((c_6 + c_7 S)S \\ + ((c_8 + c_9\theta)\theta + c_{10}))S + ((((c_{11} + c_{12}\theta)\theta + c_{13})\theta + c_{14}))S\end{aligned} \tag{3}$$

is somewhat faster than (2), especially when compiler optimizations are not used for their maximum strength (see results shown later).

There are other ways to reduce calculations. ROMS uses a Jacket and McDougal 1995 algorithm, a nested polynomial fit, where each coefficient of the final polynomial is itself computed as a polynomial. While the ROMS implementation could presumably be improved, in its present form which includes so many terms, it will be much more difficult to optimize it to the point where it will be competitive with (2) or (3), compared with the more straight forward POM's algorithm in (1).

3. Performance tests

To demonstrate the sensitivity of the calculations to the form of the EOS we performed simple experiments, calculating a three dimensional density field over 101x101x21 grid points and 1000 time steps, using a single processor. Since the size of each ocean model code is very different (e.g., the modular ROMS code has ~20 times more lines of code than the simpler POM code, see the model comparison in Ezer et al., 2002), only the EOS routine (without the pressure term) is executed. Double precision is used in all cases.

The experiments include 4 different EOS codes labeled as follows:

ROMS- the EOS in the ROMS code, which is based on Jacket and McDougal (1995).

POM- the EOS in the POM code, which is based on Mellor (1991) and written in the form of (1).

UNOPT- optimized UNESCO code where the terms in the original formulation rearranged in the form of (2).

DKAP7- the new polynomial fit (named for D. Kruger-A. Pence, version 7) in the form of (3).

Without the pressure effect the ROMS, POM and UNOPT EOS codes give identical densities to the values obtained by the original UNESCO formula (up to computer round off errors). The new fit in the DKAP7 code gives slightly different results, but the difference between UNESCO and DKAP7 is only ~10^{-6} kg m^{-3} (Table 1) so the error may not be significant for most practical applications.

Each of the 4 algorithms was executed on 3 different platforms and 4 different compilers, representing a range of computers and operation systems: SGI supercomputer cluster (but using only one of its processors out of the 100s available), Dell workstation (tested with two different compilers), and Dell laptop. Table 2 summarizes the attributes of the 4 different platforms and compilers that were used for each of the 4 codes (for a total of 4 codes x 4 platforms x 2 compilation options = 32 experiments). On each of the above platform two experiments were performed, one without compiler optimization and one with optimization option such as "–O3" or "–fast", so that the speedup achieved by rewriting the code can be compared with the speedup achieved by the compiler itself (which does not require any change in existing ocean models codes). The comparison is summarized in Fig. 1. The results demonstrate that on all platforms rewriting the

calculations (as in UNOPT or DKAP7 codes) can improve the efficiency beyond what is possible with compilers optimization alone. However, with good compiler optimization such as the SGI f90 (Fig. 1a), DKAP7 is faster than POM and ROMS only by factors of 2 and 3, respectively, while on less efficient compilers such as GNU g77 (Fig. 1d), DKAP7 is faster than POM and ROMS by as much as factors of 16 and 6, respectively. On two platforms (Windows and SGI) the new polynomial code was so well optimized that the compiler optimizers were unable to achieve any further improvement. The ROMS code was more efficient that the POM code on all platforms except the SGI; this may be explained by the additional penalty in the multiprocessor SGI for writing more temporary coefficients in ROMS. Additional experiments (not shown) running the codes using multiprocessors on the SGI cluster (using its automatic parallelization) indicate that the relative speedup achieved on a single processor carries on to the parallel codes.

The main point of this comparison is to demonstrate that codes that produce practically identical numerical results can still differ in their computational costs by an order of magnitude, indicating the great potential to improve the efficiency of ocean models (and other codes).

4. Summary and conclusions

The search for the perfect Equation of State will undoubtedly go on for a long time to come. Here, a new fit that takes into account computational considerations appear to significantly improve the speed of the equation of state in computer models. While it is difficult to characterize the speed of an algorithm in general for different architectures, our new fit for the equation of state is 6-16 times faster on a PC than the existing POM implementation, and the fastest on every architecture on which it was tested. However, it was also shown that even rewriting the equations currently used in existing models, can achieve almost as much improvement as the new fits. The results indicate that if programmers do not pay much attention to the basic code structure, the performance of codes are very machine-dependent (Fig. 1) where it is difficult to predict how the same code will perform on a particular computer or even on the same computer when using different compilers.

Beyond the improvement of the EOS itself, this study has further implications on the efficiency of ocean models in general. In the past, high resolution and large scale ocean (and atmospheric) models could only be executed on large supercomputers that

were available only at national labs and universities, so codes were not particularly optimized for smaller PCs. However, with the dramatic increase in recent years in memory and speed of PCs, more users run ocean models on local PCs. This study demonstrates the potential to significantly improve the performance of ocean models and thus allow higher resolution models than previously possible. A follow up study now underway will evaluate in more details the accuracy and speed of the pressure dependent terms and the possible influence that the EOS may have on realistic model calculations. A prototype of new ocean model codes is under development that will speedup other parts of the code in similar manner as demonstrated here with the EOS.

Acknowledgments

The authors are indebted to Martin Senator, Roger Pinkham, Peter Stahley and George Mellor for help and advice. T.E. is supported by ONR grant N00014-04-10381, and by MMS and NSF grants. A.B. was funded under ONR grant, N00014-03-1-0633.

References

Blumberg, A.F., Mellor, G.L., 1987. A description of a three-dimensional coastal ocean circulation model. In: Three-Dimensional Coastal ocean Models, edited by N. Heaps, American Geophysical Union, 1-16.

Blumberg, A.F., Signell, R.P., Jenter, H.L., 1993. Modeling transport processes in the coastal ocean, Journal of Marine Environmental Engineering, 1, 31-52.

Bryan, K., Cox, M.D., 1972. An appropriate equation of state for the study of the circulation of the world ocean. Journal of Physical Oceanography, 2, 319-335.

Ezer, T., Mellor, G.L., 2004. A generalized coordinate ocean model and a comparison of the bottom boundary layer dynamics in terrain following and in z-level grids. Ocean Modelling, 6, 379-403.

Ezer, T., Arango, H., Shchepetkin, A.F., 2002. Developments in terrain-following ocean models: Intercomparisons of numerical aspects. Ocean Modelling, 4, 249-267.

Ezer, T. and Mellor, G.L., 1997. Simulations of the Atlantic Ocean with a free surface sigma coordinate ocean model. Journal of Geophysical Research, 102(C7): 15,647-15,657.

Fofonoff, N.P., 1956. Some properties of sea water influencing the formation of Antarctic bottom water. Deep-Sea Research, 4(1), 32-35.

Haidvogel, D.B., Arango, H.G., Hedstrom, K., Beckmann, A., Malanotte- Rizzoli, P., Shchepetkin, A.F., 2000. Model evaluation experiments in the North Atlantic basin: Simulations in nonlinear terrain-following coordinates, Dynamics of Atmospheres and Oceans, 32, 239-381.

Jackett, D.R., McDougall, T.J., 1995. Minimal adjustment of hydrostatic profiles to achieve static stability. Journal of Atmospheric and Oceanic Technology, 12, 381-389.

Knuth, D.E., 1973. The art of computer programming. Addison-Wesley, 467 pp.

McDougall, T.J., Jackett, D.R., Wright, D.G., Feistel R., 2003. Accurate and computationally efficient algorithms for potential temperature and density of seawater. Journal of Atmospheric and Oceanic Technology, 20, 730-741.

Mellor, G. L., 1991. An equation of state for numerical models of ocean and estuaries. Journal of Atmospheric and Oceanic Technology, 8, 609-611.

Mellor, G.L., Yamada, T., 1982. Development of a turbulent closure model for geophysical fluid problems. Review of Geophysics, 20, 851-875.

Shchepetkin, A.F., McWilliams, J.C., 2005. The Regional Oceanic Modeling System (ROMS): A split-explicit, free surface, topography-following-coordinate oceanic model. Ocean Modelling, In Press.

UNESCO, 1981. Tenth report of the Joint panel on oceanographic tables and standards. Technical Report in Marine Science No. 36, UNESCO, Paris, 25 pp.

Table 1. Comparison of the density calculated from the DKAP7 and the UNESCO formulations for typical ocean temperature and salinity values. The units are temperature, T, in °C, salinity, S, in parts per thousand (ppt), and density relative to reference dendity, ρ-1000, in kg m^{-3}.

T	S	ρ_{DKAP7}	ρ_{UNESCO}	ρ_{DKAP7}- ρ_{UNESCO}
5	33	26.0913	26.0900	0.0013
5	35	27.6770	27.6755	0.0016
5	37	29.2678	29.2622	0.0016
10	33	25.3910	25.3909	0.0001
10	35	26.9528	26.9524	0.0004
10	37	28.5159	28.5153	0.0006
15	33	24.4403	24.4312	-0.0009
15	35	25.9721	25.9728	-0.0007
15	37	27.5154	27.5159	-0.0004
20	33	23.2374	23.2382	-0.0008
20	35	24.7625	24.7630	-0.0005
20	37	26.2893	26.2895	-0.0002
25	33	21.8323	21.8324	0.0000
25	35	23.3434	23.3431	0.0004
25	37	24.8562	24.8555	0.0007
30	33	20.2294	20.2301	-0.0007
30	35	21.7283	21.7286	-0.0003
30	37	23.2289	23.2290	-0.0001

Table 2. Attributes of the different computers and compilers used.

Computer type	Operating System	Memory	Clock speed	Fortran compiler	Compiler optimization
SGI Origin 3800	IRIX	1 GB	600 MHz	SGI f90	-Ofast
Dell workstation	Linux	1 GB	2.8 GHz	GNU g77	-O3
Dell workstation	Linux	1 GB	2.8 GHz	Portland pgf90	-fast
Dell laptop	Windows	256 MB	400 MHz	Digital f90	-fast

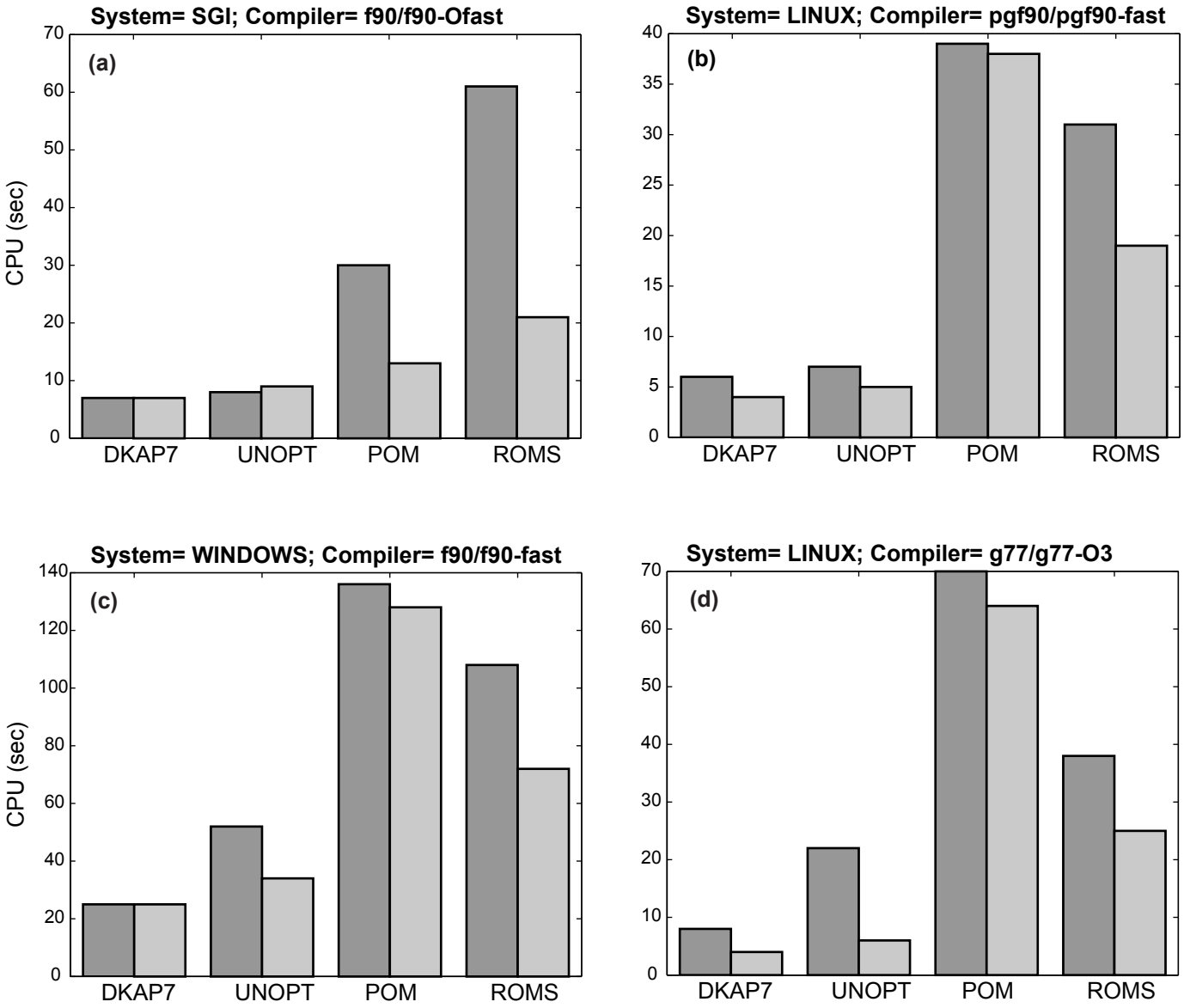

Figure 1. Code performance comparison for four codes of the Equation of State (see text for detail). The result is in seconds for 1000 time steps running on four platforms: (a) SGI computer with f90 compiler, (b) Linux workstation with pgf90 compiler, (c) Windows laptop with f90 compiler and (d) as (b), but with g77 compiler; note the different scale of each panel. The dark shaded bar graphs represent runs with no compiler optimization and the light shaded bar graphs represent runs with compiler optimization as indicated.