CNRS - Université Pierre et Marie Curie - Université Versailles-Saint-Quentin
CEA - ORSTOM - Ecole Normale Supérieure - Ecole Polytechnique

Institut Pierre Simon Laplace

des Sciences de l'Environnement Global

Notes du Pôle de Modélisation

OPA 8.1
Ocean General Circulation Model
Reference Manual

Gurvan Madec, Pascale Delecluse,
Maurice Imbard et Claire Lévy

Laboratoire d'Océanographie DYnamique et de Climatologie

Décembre 1998 **Note n° 11**

OPA 8.1 Ocean General Circulation Model Reference Manual

Gurvan Madec, Pascale Delecluse, Maurice Imbard et Claire Lévy

Laboratoire d'Océanographie DYnamique et de Climatologie
CNRS/ORSTOM/UPMC, UMR 7617

ABSTRACT

OPA is a primitive equation model of both the regional and global ocean circulation. It is intended to be a flexible tool for studying ocean and its interactions with the others components of the earth climate system (atmosphere, sea-ice, biogeochemical tracers, ...) over a wide range of space and time scale. Prognostic variables are the three-dimensional velocity field and the thermohaline variables. The distribution of variables is a three-dimensional Arakawa-C-type grid using prescribed *z*- or *s*-levels. Various physical choices are available to describe ocean physics, including a 1.5 turbulent closure for the vertical mixing. OPA is interfaced with a sea-ice model, a passive tracer model and, via the OASIS coupler, with several atmospheric general circulation models. In addition, it can be run on many different computers, including shared and distributed memory multi-processor computers.

RÉSUMÉ

OPA est un modèle aux équations primitives de la circulation océanique régionale et globale. Il se veut un outil flexible pour étudier sur un vaste spectre spatio-temporel l'océan et ses interactions avec les autres composantes du système climatique terrestre (atmosphère, glace de mer, traceurs biogéochimiques, ...). Les variables pronostiques sont le champ tri-dimensionnel de vitesse et les caractéristiques thermohalines de l'eau de mer. La distribution des variables se fait sur une grille *C* d'Arakawa tri-dimensionnelle utilisant des niveaux *z* ou *s*. Différents choix sont proposés pour décrire la physique océanique, incluant notamment une fermeture turbulente d'ordre 1.5 pour le mélange vertical. OPA est interfacé avec un modèle de glace de mer, un modèle de traceur passif et, via le coupleur OASIS, à plusieurs modèles de circulation générale atmosphérique. En outre, il peut être exécuté sur de nombreux calculateurs, y compris des machines multi-processeurs à mémoire partagée ou distribuée.

DISCLAIMER

OPA (an acronym for "Océan PArallélisé"), the Ocean General Circulation Model (OGCM) developed at the Laboratoire d'Océanographie DYnamique et de Climatologie (LODYC), is intended to be a flexible tool for studying the ocean and its interactions with the others components of the earth climate system (atmosphere, sea-ice, chemical tracers, ...) over a wide range of space and time scales. It is first and foremost an ocean modelling research tool used by researchers and students. The model and the reference manual have been made available as a service to the climate and oceanographic community. We cannot certify that the code and its manual are free of errors. Bug are inevitable and some have undoubtedly survived the testing phase. Researchers are encouraged to bring them to our attention. Anyone may use OPA freely for research purposes. The authors and the LODYC assume no responsibility for problems, errors, or incorrect usage of OPA. The researchers clearly accept full responsibility that their particular configuration is working correctly.

The OPA OGCM reference in papers and other publications is as follows:
Madec, G., P. Delecluse, M. Imbard, and C. Lévy, 1998: OPA 8.1 Ocean General Circulation Model reference manual. *Note du Pôle de modélisation*, Institut Pierre-Simon Laplace (IPSL), France, N□ 11 , 91pp.

Gurvan Madec: gm@lodyc.jussieu.fr
Pascale Delecluse: pna@lodyc.jussieu.fr
Maurice Imbard: mi@lodyc.jussieu.fr
Claire Lévy: elle@lodyc.jussieu.fr

FOREWORD

This manual presents OPA, the Ocean General Circulation Model (OGCM) which has been developed at the Laboratoire d'Océanographie DYnamique et de Climatologie (LODYC) to study large scale ocean circulation and its interaction with the atmosphere and the sea-ice. The general philosophy consists in solving the primitive equations on powerful computers with various physical choices. Although these equations are well established, different choices can be made for the physics or the algorithms. The art of numerical modelling consists in trying to choose the best parameterizations and the most efficient algorithms on a given computer to study a particular problem.

An OGCM is in perpetual evolution, so its description has to be updated regularly. The present manual describes the release 8.1 of OPA. The developments for a new computer architecture and the addition of new physics have motivated this release. The major modifications are (1) the adaptation to distributed memory computers (such as Cray T3E) using message passing methods, (2) the introduction of a terrain-following vertical coordinate (*s*-coordinates), (3) the interfacing of the model with AGCMs through OASIS 2.0 coupler [Terray 1996], a sea-ice model, and a passive tracer model (on- or off-line) (4) a rotation of the lateral diffusive and viscous tensors along geopotential surfaces or local isopycnal surfaces (neutral surfaces), (5) the introduction of the Gent and McWilliams [1990] parameterization of eddy induced velocity, (6) a linear or quadratic bottom friction, (7) a new formulation of the UNESCO equation of state and of the local Brunt-Vaisälä frequency, and (8) an additional parameterization of convective processes. In addition, several minor modifications in the coding have been introduced (rewriting of the initialisations, of some of the internal routines, ...) with the constant concern of reducing the in core memory requirement. Last but not least, phasing in the adjoint and tangent linear version of OPA has been ensured.

INTRODUCTION

Oceans cover 70% of the earth's surface and contain 97% of the earth's water. They are an essential element of our life and environment. They play a paramount role in the climate system through complex air-sea interactions and through their huge storage capacity of heat and dissolved gases (CO_2, CFC, ...). Dynamical oceanography is a young science. The detailed structure of the currents, the water mass displacements, and the distribution of physical and chemical properties in the sea are far from being well understood. The physical processes driving the currents and determining physical properties of sea water are numerous, complex and occur over a large spectra of space and time scales. They result from the circulation induced by the action of the wind on the sea surface and from the circulation linked to the spatial heterogeneities of temperature and salinity (generated at the sea surface by interactions with the atmosphere and the sea ice), called the thermohaline circulation.

Understanding and simulating the ocean has some similarities with the problem of weather forecasting. This is probably why, historically, many techniques used in the meteorological field have been applied to the oceanographic problem. In particular, for over twenty years, the use of computers to solve the Navier-Stokes equations has been successfully applied to the ocean, and Ocean General Circulation Models (OGCMs) have become important tools in the study of the dynamics and the physics of the ocean. Coupled with Atmospheric General Circulation Models (AGCMs) and/or a sea-ice model they can also be used to understand the climate evolution and hopefully to predict it.

Most OGCMs are based on the equations described by K. Bryan [1969]. They have been used to study different physical phenomena such as turbulent eddies in oceanic mid latitudes [Cox and Bryan 1984], the thermohaline circulation [Bryan 1987], meridional heat fluxes [Bryan 1982] and the general circulation of the tropical oceans [Philander and Pacanowski 1986]. It is worth noting that most of the pioneering simulations have been made in the seventies in the United States where the most powerful computers available were located at that time. Since then, countries like Great Britain, Germany, and France have also developed OGCMs, as soon as powerful systems became available in Europe.

The basic idea of numerical methods consists in discretizing differential equations on a three dimensional grid and computing the time evolution of each variable for each gridpoint. Ocean models are usually written in finite difference form. Such a method provides a legible computer code*, easy to update, and is able to deal with the complex boundary conditions formed by the coastline geometry and the bottom topography. Despite the similarities of the equations governing the ocean and the atmosphere, there is a large difference between air and sea water densities so that the characteristic scales in time and space of these two fluids are rather different. For instance, oceanic structures are characterised by highs and lows propagating across a turbulent fluid, but the time scale of these motions is about one month as compared to two or three days in the atmosphere. Moreover, the horizontal resolution needed to resolve the dynamics correctly must be of the order of the internal radius of deformation, which is *~1000* km in the atmosphere and *~30* km in the ocean. This difference is of particular importance in the study of the circulation on a global scale. The number of gridpoints in the horizontal plane is considerably larger in an ocean model. The number of vertical levels is similar for oceanic general circulation models and for atmospheric general circulation models. For the earth's domain, this results in about *160,000* x *30* gridpoints in an OGCM compare to *4000* x *20* in an AGCM. In order to ensure the numerical stability, the time step is *~10* minutes for the atmosphere and *1* hour for the ocean for the same time differencing scheme. however, the number of operations and variables per gridpoint is slightly greater for the atmosphere due to the inclusion of physical processes such as cloud physics.

The present manual describes the release 8.1 of OPA. The earliest code was developed and implemented for a Cray 1 by M. Chartier [**1985****], in collaboration with P. Delecluse. It was successfully used in 1986 to simulate the tropical Atlantic. In 1988, the model has been entirely rewritten to be used in a multitasked way and running in central memory, and named OPA for "Océan PArallelisé" [Andrich *et al.* **1988a**, **1988b**, Andrich **1989**]. A series of releases (from 2 to 5) were the occasion of some improvements on the physics (inclusion of salinity, realistic heat fluxes, initialisation

** The legibility of a computer code is of paramount importance as a model is an evolutive research tool, that has to be used by research scientists as well as students.*

*** A year in bold indicates that the reference can be found at the end of the manual in part "OPA-reference".*

with the Levitus data set) [Madec **1990**] and on the numerics (solution of the barotropic equation with a preconditioned conjugate gradient method, different treatments of static instabilities, iterative method to compute the vertical diffusion) [Madec *et al.* **1988**, **1991**]. OPA 6.0 was developed in 1990 for global ocean configuration [Marti **1992**]. The major additions were the inclusion of variable bottom topography and islands, the generalisation of the curvilinear formalism and cyclic boundary conditions [Madec and Marti **1990**, Marti *et al.* **1992**]. In 1992, the code was subjected to a major rewriting which lead to OPA 7.0: the model was rewritten in order to provide one routine for each term in the momentum and tracer equations, to introduce new coding rules, and to use UNIX facilities like cpp. In addition, implicit treatment of vertical diffusion and a 1.5 turbulent closure were introduced [Blanke **1992**, Blanke and Delecluse **1993**]

Various applications have been performed with the code, from process studies in the Mediterranean Sea [Madec *et al.* **1991**, **1996**, Speich *et al.* **1996**, Mortier **1992**, Herbaut **1994**, Herbaut *et al.* **1996**, **1997**, **1998**] to basin scale studies in the tropical Atlantic Ocean [Merle and Morlière **1988**, Morlière **1989**, Morlière *et al.* **1989**, Morlière and Duchène **1990**, Reverdin *et al.* **1991**, Blanke **1992**, Blanke and Delecluse **1993**, Delecluse *et al.* **1994**], the tropical Pacific ocean [Dandin **1993**, Boulanger **1994**, Maes *et al*. **1997**], the three tropical oceans [Maes **1996**, Boulanger *et al.* **1997**, Maes *et al.* **1998**] and the global ocean [Marti **1992**, Delecluse **1993**, Madec and Imbard **1996**, Aumont *et al.* **1998a**, **1998b**] as well as coupled studies with biogeochemical model [Lévy et al **1997**, **1998**, Stoens *et al.* **1998a**, **1998b**] and with ocean-atmosphere and sea-ice models [Mechoso *et al.* **1994**, Terray *et al.* **1995**, Guilyardi *et al.* **1995**, Terray **1996**, Guilyardi and Madec **1997**, Vintzileos and Sadourny **1997**, Vintzileos *et al.* **1998a**, **1998b**, Barthelet *et al.* **1998**, Guilyardi *et al.* **1998**, Delecluse **1998**] and adjoint studies [Greiner **1993**, Greiner and Périgaud **1994**, Greiner *et al.* **1998a**, **1998b**]. In addition, the adaptation of OPA7 to massively parallel computers has been achieved [Guyon **1995**, Guyon *et al.* **1994**, **1999**]. For a more exhaustive bibliography on studies using the model and/or its outputs, see the OPA-References in the present manual.

This manual is organised in four parts. The first part presents the model basics, i.e. the equations and their assumptions, the two system of vertical coordinate used (*z*- and *s*-coordinates), and the subgrid scale physics. The second part details the time and space discretizations. The third part is devoted to the model physics and provides the physical basis and the numerical implementation of the different options offered in the model. Finally the fourth part describes code aspects (architecture, flow trace, parallelization, environment).

All the namelist parameters and cpp keys used are referenced in the course of the manual and summerized in an index. The definition of these variables can be found in chapter IV. A nearly complet list of papers, Phd dissertations and repports which use OPA or its outputs is given in part "OPA-reference". A year in bold in the manual indicates that the reference can be found at the end of the manual in part "OPA-reference"

References

(see OPA-Bibliography when the year is in **bold**)

Bryan F., 1987: Parameter sensitivity of Primitive Equation Ocean General Circulation Models, *J. Phys. Oceanogr.*, *17*, 970-985.

Bryan, K., 1969: A numerical method for the study of the circulation of the world ocean. *J. Comput. Phys., 4*, 347-379.

———, 1982: Poleward heat transport by the ocean, Observations and models, *Annu. Rev. Earth Planet. Sci.*, *10*, 15-38.

Cox M. J., and K. Bryan, 1984: A numerical model of the ventilated thermocline, *J. Phys. Oceanogr.*, *14*, 674-687.

Pacanowski R. C., and S. G. H. Philander, 1981: Parameterization of vertical mixing in numerical models of tropical oceans. *J. Phys. Oceanogr.*, *11*, 1443-1451.

Philander, S. G. H., and R. C. Pacanowski, 1986: A model of the seasonal cycle in the tropical Atlantic Ocean. *J. Geophys. Res.*, *91*, 14,192-14,206.

I. MODEL BASICS

I.1 PRIMITIVE EQUATIONS

I.1-a Vector Invariant Formulation

The ocean is a fluid which can be described to a good approximation by the primitive equations, i.e. the Navier-Stokes equations along with a non-linear equation of state which couples the two active tracers (temperature and salinity) to the fluid velocity, plus the following additional assumptions made from scale considerations :

(1) spherical earth approximation: the geopotential surfaces are assumed to be spheres so that gravity (local vertical) is parallel to the earth's radius ;

(2) thin-shell approximation: the ocean depth is neglected compared to the earth's radius ;

(3) turbulent closure hypothesis: the turbulent fluxes (which represent the effect of small scale processes on the large-scale) are expressed in terms of large-scale features ;

(4) Boussinesq hypothesis: density variations are neglected except in their contribution to the buoyancy force ;

(5) Hydrostatic hypothesis: the vertical momentum equation is reduced to a balance between the vertical pressure gradient and buoyancy force (this removes convective processes from the initial Navier-Stokes equations: they must be parameterized) ;

(6) Incompressibility hypothesis: the three dimensional divergence of the velocity vector is assumed to be zero.

Because the gravitational force is so dominant in the equations of large-scale motions, it is quite useful to choose an orthogonal set of unit vectors $(\mathbf{i}, \mathbf{j}, \mathbf{k})$ linked to the earth such that $\mathbf{k}$ is the local upward vector and $(\mathbf{i}, \mathbf{j})$ are two vectors orthogonal to $\mathbf{k}$, i.e. tangent to the geopotential surfaces. Let us define the following variables: $\mathbf{U}$ the vector velocity, $\mathbf{U} = \mathbf{U}_h + w\,\mathbf{k}$ (the subscript h denotes the local horizontal vector, i.e. over the $(\mathbf{i}, \mathbf{j})$ plan), T the potential temperature, S the salinity, ρ the *in-situ* density. The vector invariant form of the primitive equations in the $(\mathbf{i}, \mathbf{j}, \mathbf{k})$ vector system provides the following six equations (namely the momentum balance, the hydrostatic equilibrium, the incompressibility, the heat and salt conservation and an equation of state):

$$\frac{\partial \mathbf{U}_h}{\partial t} = -\left[(\nabla \times \mathbf{U}) \times \mathbf{U} + \frac{1}{2} \nabla (\mathbf{U}^2) \right]_h - f\,\mathbf{k} \times \mathbf{U}_h - \frac{1}{\rho_o} \nabla_h p + \mathbf{D}^{\mathbf{U}} \tag{I.1.1}$$

$$\frac{\partial p}{\partial z} = -\rho\, g \tag{I.1.2}$$

$$\nabla \cdot \mathbf{U} = 0 \tag{I.1.3}$$

$$\frac{\partial T}{\partial t} = -\nabla \cdot \left(T\,\mathbf{U} \right) + D^T \tag{I.1.4}$$

$$\frac{\partial S}{\partial t} = -\nabla \cdot \left(S\,\mathbf{U} \right) + D^S \tag{I.1.5}$$

$$\rho = \rho(T, S, p) \tag{I.1.6}$$

where ∇ is the generalised derivative vector operator in $(\mathbf{i}, \mathbf{j}, \mathbf{k})$ directions, t the time, z the vertical coordinate, ρ the *in situ* density given by the equation of state (I.1.6), ρ_o a reference density, p the pressure, f the Coriolis acceleration ($f = 2\,\Omega.\mathbf{k}$, where Ω is the Earth angular velocity vector), and g the gravitational acceleration. $\mathbf{D}^{\mathbf{U}}$, D^T and D^S are the parameterizations of small scale physics for momentum, temperature and salinity, including surface forcing terms. Their nature and formulation are discussed in § I.5.

I.1-b Boundary Conditions

An ocean is bounded by complex coastlines and bottom topography at its base and by an air-sea or ice-sea interface at its top. These boundaries can be defined by two surfaces, $z = -H(i, j)$ and $z = \eta(i, j, t)$, where H is the depth of the ocean bottom and η the height of the sea surface. Both H and η are usually referenced to a given surface, $z = 0$, chosen as a mean sea surface (Fig. I.1). Through these two boundaries, the ocean can exchange fluxes of heat, fresh water, salt, and momentum with the solid earth, the continental surfaces, the sea ice and the atmosphere. However, some of these fluxes are so weak that even on climatic time scales of thousands of years they can be neglected. In the following, we briefly review

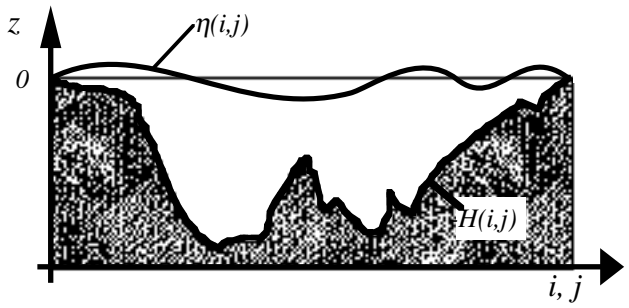


Figure I.1: The ocean is bounded by two surfaces, $z = -H(i,j)$ and $z = \eta(i,j,t)$, where H is the depth of the sea floor and η the height of the sea surface. Both H and η are referenced to $z = 0$.

the fluxes exchanged at the interfaces between the ocean and the other components of the earth system.

- *Land - ocean interface*: the major flux between continental surfaces and the ocean is a mass exchange of fresh water through river runoff. Such an exchange modifies locally the sea surface salinity especially in the vicinity of major river mouths. It can be neglected for short range integrations but has to be taken into account for long term integrations as it influences the characteristics of water masses formed (especially at high latitudes). It is required to close the water cycle of the climatic system. It is usually specified as a fresh water flux at the air-sea interface in the vicinity of river mouths.

- *Solid earth - ocean interface*: heat and salt fluxes across the sea floor are negligibly small, except in special areas of little extent. They are always neglected in the model. The boundary condition is thus set to no flux of heat and salt across solid boundaries. For momentum, the situation is different. There is no flow across solid boundaries, i.e. the velocity normal to the ocean bottom and coastlines is zero (in other words, the bottom velocity is parallel to solid boundaries). This kinematic boundary condition can be expressed as:

$$w = -\mathbf{U}_h . \nabla_h (H) \tag{I.1.7}$$

In addition, the ocean exchanges momentum with the earth through friction processes. Such momentum transfer occurs at small scales in a boundary layer. It must be parameterized in terms of turbulent fluxes through bottom and/or lateral boundary conditions. Its specification depends on the nature of the physical parameterization used for $\mathbf{D}^{\mathbf{U}}$ in (I.1.1). They are discussed in § I.5 and § III.6 to 9.

- *Atmosphere - ocean interface*: the kinematic surface condition plus the mass flux of fresh water *P-E* (the precipitation minus evaporation budget) leads to:

$$w = \frac{\partial \eta}{\partial t} + \mathbf{U}_h \big|_{z=\eta} . \nabla_h (\eta) + P - E \tag{I.1.8}$$

The dynamic boundary condition, neglecting the surface tension (which removes capillary waves from the system) leads to the continuity of pressure across the interface $z = \eta$. The atmosphere and ocean also exchange horizontal momentum (wind stress), and heat.

- *Sea ice - ocean interface*: the two media exchange heat, salt, fresh water and momentum. The sea-surface temperature is constrained to be at the freezing point at the interface. Sea ice salinity is very low (~4 *psu*) compared to those of the ocean (~34 *psu*). The cycle of freezing/melting is associated with fresh water and salt fluxes that cannot be neglected.

I.2 THE HORIZONTAL PRESSURE GRADIENT

I.2-a Pressure Formulation

The total pressure at a given depth *z* is composed of a surface pressure p_s at a reference geopotential surface ($z = 0$) and a hydrostatic pressure p_h such that: $p(i,j,z,t) = p_s(i,j,t) + p_h(i,j,z,t)$. The latter is computed by integrating (I.1.2), assuming that pressure in decibars can be approximated by depth in meters in (I.1.6). The hydrostatic pressure is then given by:

$$p_h(i,j,z,t) = \int_{\varsigma=z}^{\varsigma=0} g\, \rho(T,S,z)\, d\varsigma \tag{I.2.1}$$

The surface pressure requires a more specific treatment. Two strategies can be considered: (1) the introduction of a new variable η, the free-surface elevation, for which a prognostic equation can be established and solved ; (2) the assumption that the ocean surface is a rigid lid, on which the pressure (or its horizontal gradient) can be diagnosed. When the former strategy is used, a solution of the free-surface elevation consists in the excitation of external gravity waves. The flow is barotropic and the surface moves up and down with gravity as the restoring force. The phase speed of such waves is high (some hundreds of metres per second) so that the time step would have to be very short if they were present in the model. The latter strategy filters these waves as the rigid lid approximation implies $\eta = 0$, i.e. the sea surface is the surface $z = 0$. This well-known approximation increases the surface wave speed to infinity and modifies certain other long-wave dynamics (e.g. barotropic Rossby or planetary waves). In the present release of OPA, only the second strategy is available.

I.2-b Diagnosing the Surface Pressure Gradient

We assume that the ocean surface ($z = 0$) is a rigid lid on which a pressure p_s is exerted. This implies that the vertical velocity at the surface is equal to zero. From the continuity equation (I.1.3) and the kinematic condition at the bottom (I.1.7) (no flux across the bottom), it can be shown that the vertically integrated flow $H\overline{\mathbf{U}}_h$ is nondivergent (where the overbar indicates a vertical average over the whole water column, i.e. from $z = -H$, the ocean bottom, to $z = 0$, the rigid-lid). Thus, $H\overline{\mathbf{U}}_h$ can be derived from a volume transport streamfunction ψ:

$$\overline{\mathbf{U}}_h = \frac{1}{H}\left(\mathbf{k}\times\nabla\psi\right) \tag{I.2.2}$$

As p_s does not depend on depth, its horizontal gradient is obtained by forming the vertical average of (I.1.1) and using (I.2.2):

$$\frac{1}{\rho_o}\nabla_h p_s = \overline{\mathbf{M}} - \frac{\partial\overline{\mathbf{U}}_h}{\partial t} = \overline{\mathbf{M}} - \frac{1}{H}\left[\mathbf{k}\times\nabla\left(\frac{\partial\psi}{\partial t}\right)\right] \tag{I.2.3}$$

Here $\mathbf{M} = (M_u, M_v)$ represents the collected contributions of the Coriolis, hydrostatic pressure gradient, non-linear and viscous terms in (I.1.1). The time derivative of ψ is the solution of an elliptic equation which is obtained from the vertical component of the curl of (I.2.3):

$$\left[\nabla\times\left[\frac{1}{H}\mathbf{k}\times\nabla\left(\frac{\partial\psi}{\partial t}\right)\right]\right]_z = \left[\nabla\times\overline{\mathbf{M}}\right]_z \tag{I.2.4}$$

Using the proper boundary conditions, (I.2.4) can be solved to find $\partial\psi/\partial t$ and thus using (I.2.3) the horizontal surface pressure gradient. It should be noted that p_s can be computed by taking the divergence of (I.2.3) and solving the resulting elliptic equation. Thus the surface pressure is a diagnostic quantity which can be recovered for analysis purposes.

I.2-c Boundary Conditions

A difficulty lies in the determination of the boundary condition on $\partial\psi/\partial t$. The boundary condition on velocity is that there is no flow normal to a solid wall, i.e. the coastlines are streamlines. Therefore (I.2.4) is solved with the following Dirichlet boundary condition: $\partial\psi/\partial t$ is constant along each coastline of the same continent or of the same island. When all the coastlines are connected (there are no islands), the constant value of $\partial\psi/\partial t$ along the coast can be arbitrarily chosen to be zero. When islands are present in the domain, the value of the barotropic streamfunction will generally be different for each island and for the continent, and will vary with respect to time. So the boundary condition is: $\psi = 0$ along the continent and $\psi = \mu_n$ along island n ($1 \le n \le Q$), where Q is the number of islands present in the domain and μ_n is a time dependent variable. A time-evolution equation of the unknown μ_n can be found by evaluating the circulation of the time derivative of the vertical average (barotropic) velocity field along a closed contour around each island. Since the circulation of a gradient field along a closed contour is zero, from (I.2.3) we have:

$$\oint_n \frac{1}{H}\left[\mathbf{k}\times\nabla\left(\frac{\partial\psi}{\partial t}\right)\right]\cdot\mathbf{d}\ell = \oint_n \overline{\mathbf{M}}\cdot\mathbf{d}\ell \qquad 1 \le n \le Q \tag{I.2.5}$$

Since (I.2.4) is linear, its solution ψ can be decomposed as follows :

$$\psi = \psi_o + \sum_{n=1}^{n=Q} \mu_n \psi_n \tag{I.2.6}$$

where ψ_o is the solution of (I.2.4) with $\psi_o = 0$ along all the coastlines, and where ψ_n is the solution of (I.2.4) with the right-hand side equal to 0, and with $\psi_n = 1$ along the island n, $\psi_n = 0$ along the other boundaries. The function ψ_n is thus independent of time. Introducing (I.2.6) into (I.2.5) yields:

$$\begin{aligned}\left[\oint_n \frac{1}{H}\left[\mathbf{k}\times\nabla\psi_m\right]\cdot\mathbf{d}\ell\right]_{\substack{1\le m\le Q\\ 1\le n\le Q}}\left(\frac{\partial\mu_n}{\partial t}\right)_{1\le n\le Q} \\ = \left(\oint_n\left[\overline{\mathbf{M}} - \frac{1}{H}\left[\mathbf{k}\times\nabla\left(\frac{\partial\psi_o}{\partial t}\right)\right]\right]\cdot\mathbf{d}\ell\right)_{1\le n\le Q}\end{aligned} \tag{I.2.7}$$

which can be rewritten as:

$$\mathbf{A}\left(\frac{\partial\mu_n}{\partial t}\right)_{1\le n\le Q} = \mathbf{B} \tag{I.2.8}$$

where $\mathbf{A}$ is a $Q\times Q$ matrix and $\mathbf{B}$ is a time-dependent vector. As $\mathbf{A}$ is independent of time, it can be calculated and inverted once. The time derivative of the stream-function when islands are present is thus given by :

$$\frac{\partial\psi}{\partial t} = \frac{\partial\psi_o}{\partial t} + \sum_{n=1}^{n=Q} \mathbf{A}^{-1}\mathbf{B}\ \psi_n \tag{I.2.9}$$

I.3 CURVILINEAR *z*-COORDINATE SYSTEM

I.3-a Tensorial Formalism

In many ocean circulation problems, the flow field has regions of enhanced dynamics (i.e. surface layers, western boundary currents, equatorial currents, or ocean fronts). The representation of such dynamical processes can be improved by specifically increasing the model resolution in these regions. As well, it may be convenient to use a lateral boundary-following coordinate system to better represent coastal dynamics. Moreover, the common geographical coordinate system has a singular point at the North Pole which cannot be easily treated in a global model without filtering. A solution consists in introducing an appropriate coordinate transformation which shifts the singular point on land [Madec and Imbard **1996**, Murray 1996]. As a conclusion, it is important to solve the primitive equations in various curvilinear coordinate systems. An efficient way of introducing an appropriate coordinate transform can be found when using a tensorial formalism. This formalism is suited to any multi-dimensional curvilinear coordinate system. Ocean modellers mainly use three-dimensional orthogonal grids on the sphere, with conservation of the local vertical. Here we give the simplified equations for this particular case. The general case is detailed by Eiseman and Stone [1980] in their survey of the conservation laws of fluid dynamics.

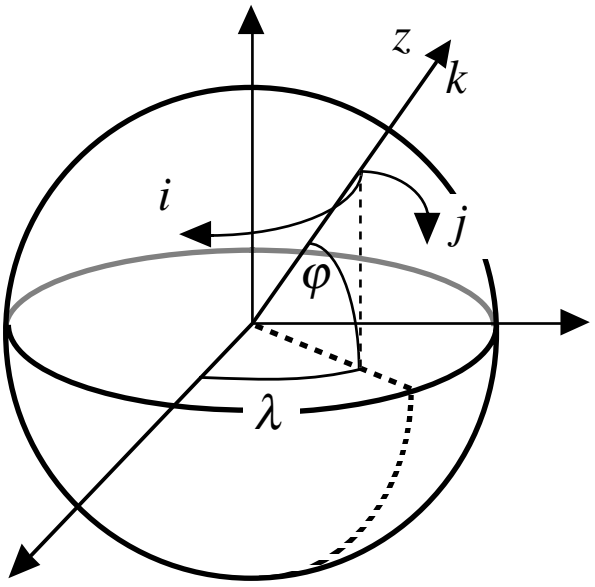


Figure I.2: the geographical coordinate system (λ, φ, z) and the curvilinear coordinate system (i, j, k).

Let (i, j, k) be a set of orthogonal curvilinear coordinates on the sphere associated with the positively oriented orthogonal set of unit vectors $(\mathbf{i}, \mathbf{j}, \mathbf{k})$ linked to the earth such that $\mathbf{k}$ is the local upward vector and $(\mathbf{i}, \mathbf{j})$ are two vectors orthogonal to $\mathbf{k}$, i.e. along geopotential surfaces (Fig. I.2). Let (λ, φ, z) be the geographical coordinates system in which a position is defined by the latitude $\varphi(i, j)$, the longitude $\lambda(i, j)$ and the distance from the centre of the earth $a + z(k)$ where a is the earth's radius and z the altitude above a reference sea level (Fig. I.1). The local deformation of the curvilinear coordinate system is given by e_1, e_2 and e_3, the three scale factors:

$$\begin{aligned} e_1 &= (a+z)\left[\left(\frac{\partial \lambda}{\partial i}\cos\varphi\right)^2 + \left(\frac{\partial \varphi}{\partial i}\right)^2\right]^{1/2} \\ e_2 &= (a+z)\left[\left(\frac{\partial \lambda}{\partial j}\cos\varphi\right)^2 + \left(\frac{\partial \varphi}{\partial j}\right)^2\right]^{1/2} \\ e_3 &= \left(\frac{\partial z}{\partial k}\right) \end{aligned} \qquad \text{(I.3.1)}$$

Since the ocean depth is far smaller than the earth's radius, $a + z$ can be replaced by a in (I.3.1) (thin-shell approximation). The resulting horizontal scale factors e_1 and e_2 are independent of k while the vertical scale factor is a single function of k as $\mathbf{k}$ is parallel to $\mathbf{z}$. The scalar and vectorial operators which appear in the primitive equations (Eqs. I.1.1 to I.1.6) can be written in the tensorial form, invariant in any orthogonal horizontal curvilinear coordinate system transformation:

$$\nabla q = \frac{1}{e_1}\frac{\partial q}{\partial i}\mathbf{i} + \frac{1}{e_2}\frac{\partial q}{\partial j}\mathbf{j} + \frac{1}{e_3}\frac{\partial q}{\partial k}\mathbf{k} \qquad \text{(I.3.2)}$$

$$\nabla . \mathbf{A} = \frac{1}{e_1 e_2}\left[\frac{\partial (e_2 a_1)}{\partial i} + \frac{\partial (e_1 a_2)}{\partial j}\right] + \frac{1}{e_3}\frac{\partial a_3}{\partial k} \qquad \text{(I.3.3)}$$

$$\begin{aligned} \nabla \times \mathbf{A} = &\left[\frac{1}{e_2}\frac{\partial a_3}{\partial j} - \frac{1}{e_3}\frac{\partial a_2}{\partial k}\right]\mathbf{i} + \left[\frac{1}{e_3}\frac{\partial a_1}{\partial k} - \frac{1}{e_1}\frac{\partial a_3}{\partial i}\right]\mathbf{j} \\ &+ \frac{1}{e_1 e_2}\left[\frac{\partial (e_2 a_2)}{\partial i} - \frac{\partial (e_1 a_1)}{\partial j}\right]\mathbf{k} \end{aligned} \qquad \text{(I.3.4)}$$

$$\Delta q = \nabla . (\nabla q) \qquad \text{(I.3.5)}$$

$$\Delta \mathbf{A} = \nabla (\nabla . \mathbf{A}) - \nabla \times (\nabla \times \mathbf{A}) \qquad \text{(I.3.6)}$$

where q is a scalar quantity and $\mathbf{A} = (a_1, a_2, a_3)$ a vector in the (i, j, k) coordinate system.

I.3-b Model Equations

In order to express the primitive equations in tensorial formalism, it is necessary to compute the horizontal component of the non linear and viscous terms of the equation using (I.3.2) to (I.3.6). Let us set $\mathbf{U} = (u, v, w) = \mathbf{U}_h + w\mathbf{k}$, the velocity in the (i, j, k) coordinate system and define the relative vorticity ζ and the divergence of the horizontal velocity field χ, by :

$$\zeta = \frac{1}{e_1 e_2}\left[\frac{\partial(e_2 v)}{\partial i} - \frac{\partial(e_1 u)}{\partial j}\right] \tag{I.3.7}$$

$$\chi = \frac{1}{e_1 e_2}\left[\frac{\partial(e_2 u)}{\partial i} + \frac{\partial(e_1 v)}{\partial j}\right] \tag{I.3.8}$$

Using the fact that horizontal scale factors e_1 and e_2 are independent of k and that e_3 is a function of the single variable k, the non-linear term of (I.1.1) can be transformed as follows :

$$\begin{aligned}
&\left[(\nabla\times\mathbf{U})\times\mathbf{U} + \frac{1}{2}\nabla(\mathbf{U}^2)\right]_h \\
&= \begin{pmatrix} \left[\frac{1}{e_3}\frac{\partial u}{\partial k} - \frac{1}{e_1}\frac{\partial w}{\partial i}\right] w - \zeta v \\ \zeta u - \left[\frac{1}{e_2}\frac{\partial w}{\partial j} - \frac{1}{e_3}\frac{\partial v}{\partial k}\right] w \end{pmatrix} + \frac{1}{2}\begin{pmatrix} \frac{1}{e_1}\frac{\partial(u^2+v^2+w^2)}{\partial i} \\ \frac{1}{e_2}\frac{\partial(u^2+v^2+w^2)}{\partial j} \end{pmatrix} \\
&= \begin{pmatrix} -\zeta v \\ \zeta u \end{pmatrix} + \frac{1}{2}\begin{pmatrix} \frac{1}{e_1}\frac{\partial(u^2+v^2)}{\partial i} \\ \frac{1}{e_2}\frac{\partial(u^2+v^2)}{\partial j} \end{pmatrix} + \frac{1}{e_3}\begin{pmatrix} w\frac{\partial u}{\partial k} \\ w\frac{\partial v}{\partial k} \end{pmatrix} \\
&\qquad - \begin{pmatrix} \frac{w}{e_1}\frac{\partial w}{\partial i} - \frac{1}{2e_1}\frac{\partial w^2}{\partial i} \\ \frac{w}{e_2}\frac{\partial w}{\partial j} - \frac{1}{2e_2}\frac{\partial w^2}{\partial j} \end{pmatrix}
\end{aligned}$$

The last term of the right hand side is obviously zero, and thus the non-linear term of (I.1.1) is written in the (i, j, k) coordinate system :

$$\begin{aligned}
&\left[(\nabla\times\mathbf{U})\times\mathbf{U} + \frac{1}{2}\nabla(\mathbf{U}^2)\right]_h \\
&\qquad = \zeta\,\mathbf{k}\times\mathbf{U}_h + \frac{1}{2}\nabla_h(\mathbf{U}_h^2) + \frac{1}{e_3}w\frac{\partial\mathbf{U}_h}{\partial k}
\end{aligned} \tag{I.3.9}$$

The equations solved by the ocean model including the rigid-lid approximation (i.e. Eqs. (I.1.1) to (I.1.6) plus Eqs. (I.2.3) and (I.1.4)) can be written in the following tensorial formalism :

* momentum equation:

$$\begin{aligned}
\frac{\partial u}{\partial t} &= +(\zeta + f)v - \frac{1}{e_3}w\frac{\partial u}{\partial k} \\
&\quad - \frac{1}{e_1}\frac{\partial}{\partial i}\left(\frac{1}{2}(u^2+v^2) + \frac{p_h}{\rho_o}\right) - \frac{1}{\rho_o e_1}\frac{\partial p_s}{\partial i} + D_u^{\mathrm{U}}
\end{aligned} \tag{1.3.10}$$

$$\begin{aligned}
\frac{\partial v}{\partial t} &= -(\zeta + f)u - \frac{1}{e_3}w\frac{\partial v}{\partial k} \\
&\quad - \frac{1}{e_2}\frac{\partial}{\partial j}\left(\frac{1}{2}(u^2+v^2) + \frac{p_h}{\rho_o}\right) - \frac{1}{\rho_o e_2}\frac{\partial p_s}{\partial j} + D_v^{\mathrm{U}}
\end{aligned} \tag{1.3.11}$$

where ζ is given by (1.3.7) and the surface pressure gradient is given by:

$$\frac{1}{\rho_o}\nabla_h p_s = \begin{pmatrix} \overline{M}_u + \frac{1}{H\,e_2}\frac{\partial}{\partial j}\left(\frac{\partial\psi}{\partial t}\right) \\ \overline{M}_v - \frac{1}{H\,e_1}\frac{\partial}{\partial i}\left(\frac{\partial\psi}{\partial t}\right) \end{pmatrix} \tag{1.3.12}$$

Here $\mathbf{M} = (M_u, M_v)$ represents the collected contributions of non-linear, viscous and hydrostatic pressure gradient terms in (1.3.10) and (1.3.11) and the overbar indicates a vertical average over the whole water column (i.e. from $z = -H$, the ocean bottom, to $z = 0$, the rigid-lid). The time derivative of ψ is the solution of an elliptic equation :

$$\begin{aligned}
&\frac{\partial}{\partial i}\left[\frac{e_2}{H\,e_1}\frac{\partial}{\partial i}\left(\frac{\partial\psi}{\partial t}\right)\right] + \frac{\partial}{\partial j}\left[\frac{e_1}{H\,e_2}\frac{\partial}{\partial j}\left(\frac{\partial\psi}{\partial t}\right)\right] \\
&\qquad = \frac{\partial}{\partial i}\left(e_2\overline{M}_v\right) - \frac{\partial}{\partial j}\left(e_1\overline{M}_u\right)
\end{aligned} \tag{1.3.13}$$

The vertical velocity and the hydrostatic pressure are diagnosed from the following equations:

$$\frac{\partial w}{\partial k} = -\chi\,e_3 \tag{I.3.14}$$

$$\frac{\partial p_h}{\partial k} = -\rho\,g\,e_3 \tag{I.3.15}$$

where χ is given by (1.3.8).

* tracer equations:

$$\frac{\partial T}{\partial t} = -\frac{1}{e_1 e_2}\left[\frac{\partial(e_2 T u)}{\partial i} + \frac{\partial(e_1 T v)}{\partial j}\right] - \frac{1}{e_3}\frac{\partial(T w)}{\partial k} + D^T \tag{I.3.16}$$

$$\frac{\partial S}{\partial t} = -\frac{1}{e_1 e_2}\left[\frac{\partial(e_2 S u)}{\partial i} + \frac{\partial(e_1 S v)}{\partial j}\right] - \frac{1}{e_3}\frac{\partial(S w)}{\partial k} + D^S \tag{I.3.17}$$

$$\rho = \rho\big(T, S, z(k)\big) \tag{I.3.18}$$

The expression of $\mathbf{D}^{\mathrm{U}}$, D^S and D^T depends on the subgrid scale parameterization used. It will be defined in § I.5.

The whole set of the continuous equations solved by the model in the *z*-coordinate systeme is summmerized in Table I.1.

References

(see OPA-Bibliography when the year is in **bold**)

Murray, R. J., 1996: Explicit generation of orthogonal grids for ocean models. *J. Comput. Phys.*, *126*, 251-273.

Eiseman, P. R., and A. P. Stone, 1980: Conservation lows of fluid dynamics - A survey. *SIAM Rev., 22*, 12-27.

$$\frac{\partial u}{\partial t} = +\left(\zeta + f\right) v - \frac{1}{e_3} w \frac{\partial u}{\partial k} - \frac{1}{2e_1} \frac{\partial}{\partial i}\left(u^2 + v^2\right)$$
$$- \frac{1}{\rho_o e_1} \frac{\partial p_h}{\partial i} - \frac{1}{\rho_o e_1} \frac{\partial p_s}{\partial i} + D_u^{\mathrm{U}} + \frac{1}{e_3} \frac{\partial}{\partial k}\left(\frac{A^{vm}}{e_3} \frac{\partial u}{\partial k} \right)$$

$$\zeta = \frac{1}{e_1 e_2}\left(\frac{\partial}{\partial i}\left[e_2 v\right] - \frac{\partial}{\partial j}\left[e_1 u\right] \right)$$

$$\chi = \frac{1}{e_1 e_2}\left(\frac{\partial}{\partial i}\left[e_2 u\right] + \frac{\partial}{\partial j}\left[e_1 v\right] \right)$$

with

$$\frac{\partial v}{\partial t} = -\left(\zeta + f\right) u - \frac{1}{e_3} w \frac{\partial v}{\partial k} - \frac{1}{2e_2} \frac{\partial}{\partial j}\left(u^2 + v^2\right)$$
$$- \frac{1}{\rho_o e_2} \frac{\partial p_h}{\partial j} - \frac{1}{\rho_o e_2} \frac{\partial p_s}{\partial j} + D_v^{\mathrm{U}} + \frac{1}{e_3} \frac{\partial}{\partial k}\left(\frac{A^{vm}}{e_3} \frac{\partial v}{\partial k} \right)$$

$$\overline{M_u} = \frac{1}{H} \int_{-H}^{0} \left[\frac{\partial u}{\partial t} + \frac{1}{\rho_o e_1} \frac{\partial p_s}{\partial i} \right] e_3 \, dk$$

$$\overline{M_v} = \frac{1}{H} \int_{-H}^{0} \left[\frac{\partial v}{\partial t} + \frac{1}{\rho_o e_2} \frac{\partial p_s}{\partial j} \right] e_3 \, dk$$

$$\frac{\partial p_h}{\partial k} = -\rho \, g \, e_3$$

$$\frac{\partial w}{\partial k} = -e_3 \, \chi$$

$$\frac{1}{\rho_o e_1} \frac{\partial p_s}{\partial i} = \overline{M_u} + \frac{1}{H e_2} \frac{\partial}{\partial j}\left(\frac{\partial \psi}{\partial t} \right)$$

$$\frac{1}{\rho_o e_2} \frac{\partial p_s}{\partial j} = \overline{M_v} - \frac{1}{H e_1} \frac{\partial}{\partial i}\left(\frac{\partial \psi}{\partial t} \right)$$

with $$\frac{\partial}{\partial i}\left[\frac{e_2}{H e_1} \frac{\partial}{\partial i}\left(\frac{\partial \psi}{\partial t} \right) \right] + \frac{\partial}{\partial j}\left[\frac{e_1}{H e_2} \frac{\partial}{\partial j}\left(\frac{\partial \psi}{\partial t} \right) \right] = \frac{\partial}{\partial i}\left(e_2 \overline{M_v} \right) - \frac{\partial}{\partial j}\left(e_1 \overline{M_u} \right)$$

$$\frac{\partial T}{\partial t} = -\frac{1}{e_1 e_2}\left[\frac{\partial}{\partial i}\left(e_2 T u\right) + \frac{\partial}{\partial j}\left(e_1 T v\right) \right] - \frac{1}{e_3} \frac{\partial}{\partial k}\left(T w\right)$$
$$+ D^{lT} + \frac{1}{e_3} \frac{\partial}{\partial k}\left(\frac{A^{vT}}{e_3} \frac{\partial T}{\partial k} \right)$$

and

$$\frac{\partial S}{\partial t} = -\frac{1}{e_1 e_2}\left[\frac{\partial}{\partial i}\left(e_2 S u\right) + \frac{\partial}{\partial j}\left(e_1 S v\right) \right] - \frac{1}{e_3} \frac{\partial}{\partial k}\left(S w\right)$$
$$+ D^{lS} + \frac{1}{e_3} \frac{\partial}{\partial k}\left(\frac{A^{vT}}{e_3} \frac{\partial S}{\partial k} \right)$$

where the expression of $\left(D_u^{lU}, D_v^{lU}\right)$ and $\left(D^{lT}, D^{lS}\right)$ is given in Table I.3 and I.4, respectively

Table I.1: Set of equations solved by the model in the curvilinear z-coordinate system.

I.4 CURVILINEAR *s*-COORDINATE SYSTEM

I.4-a Introduction

Several important aspects of the ocean circulation are influenced by bottom topography. Of course, the most important is that bottom topography determines deep ocean sub-basins, barriers, sills and channels that strongly constrain the path of water masses, but more subtle effects exist. For example, the topographic β-effect is usually larger than the planetary one along continental slopes. Topographic Rossby waves can be excited and can interact with the mean current. In the *z*-coordinate system presented in the previous section (§ I.3), *z*-surfaces are geopotential surfaces. The bottom topography is discretized by steps. This often leads to a misrepresentation of a gradually sloping bottom and to large localized depth gradients associated with large localized vertical velocities. The response to such a velocity field often leads to numerical dispersion effects.

A terrain-following coordinate system (hereafter *s*-coordinates) avoids the discretization error in the depth field since the layers of computation are gradually adjusted with depth to the ocean bottom. Relatively shallow topographic features in the deep ocean, which would be ignored in typical *z*-model applications with the largest grid spacing at greatest depths, can easily be represented (with relatively low vertical resolution) as can gentle, large-scale slopes of the sea floor. A terrain-following model (hereafter *s*-model) also facilitates the modelling of the boundary layer flows over a large depth range, which in the framework of the *z*-model would require high vertical resolution over the whole depth range. Moreover, with *s*-coordinates it is possible, at least in principle, to have the bottom and the sea surface as the only boundaries of the domain. Nevertheless, *s*-coordinates also have its drawbacks. Perfectly adapted to an homogeneous ocean, it has strong limitations as soon as stratification is introduced. The main two problems come from the truncation error in the horizontal pressure gradient and a possibly increased diapycnal diffusion. The horizontal pressure force in *s*-coordinates consists of two terms (see Appendix A),

$$\nabla p\big|_z = \nabla p\big|_s - \frac{\partial p}{\partial s}\nabla z\big|_s \tag{I.4.1}$$

The second term in (I.4.1) depends on the tilt of the coordinate surface and introduces a truncation error which is not present in a *z*-model. In the special case of σ-coordinates (i.e. a depth-normalised coordinate system $\sigma = z/H$) Haney [1991] and Beckmann and Haidvogel [1993] have given estimates of the magnitude of this truncation error. It depends on topographic slope, stratification, horizontal and vertical resolution, and the finite difference scheme. This error limits the possible topographic slopes that a model can handle at a given horizontal and vertical resolution. This is a severe restriction for large-scale applications using realistic bottom topography. The large scale slopes require high horizontal resolution, and the computational cost becomes prohibitive. This problem can be, at least partially, overcome by mixing *s*-coordinates and step-like representation of bottom topography [Madec *et al.* **1996**]. However, another problem is then raised in the definition of the model domain.

A minimum of diffusion along the coordinate surfaces of any finite difference model is always required for numerical reasons. It causes spurious diapycnal mixing when coordinate surfaces do not coincide with isopycnal surfaces. This is the case for a *z*-model as well as for an *s*-model. However, density varies more strongly on *s*-surfaces than on horizontal surfaces in regions of large topographic slopes, implying larger diapycnal diffusion in a *s*-model than in a *z*-model. Whereas such a diapycnal diffusion in a *z*-model tends to weaken horizontal density (pressure) gradients and thus the horizontal circulation, it usually reinforces these gradients in a *s*-model, creating spurious circulation. For example, imagine an isolated bump of topography in an ocean at rest with a horizontally uniform stratification. Spurious diffusion along *s*-surfaces will induce a bump of isopycnal surfaces over the topography, and thus will generate there a baroclinic eddy. In contrast, the ocean will stay at rest in a *z*-model. As for the truncation error, the problem can be reduced by introducing the terrain-following coordinate below the strongly stratified portion of the water column (i.e. the main thermocline) [Madec *et al.* **1996**]. An alternate solution consists in rotating the lateral diffusive tensor to geopotential or to isopycnal surfaces (see § I.5 and Appendix B).

The *s*-coordinates introduced here [Lott and Madec **1989**, Lott *et al.* **1990**, Madec *et al.* **1996**] differ mainly in two aspects from similar models. It combines the properties which make OPA suitable for climate applications with a good representation of bottom topography allowing mixed step-like/terrain following topography. It also offers a completely general transformation, $s = s(i, j, z)$, for the vertical coordinate which goes beyond those of previous hybrid models except the GFDL version developed by Gerdes [1993a, 1993b] which has similar properties as the OPA release presented here.

$$\frac{\partial u}{\partial t} = +(\zeta + f)\, v - \frac{1}{e_3}\,\omega\,\frac{\partial u}{\partial k} - \frac{1}{2e_1}\frac{\partial}{\partial i}\left(u^2 + v^2\right)$$
$$-\frac{1}{\rho_o e_1}\frac{\partial p_h}{\partial i} + g\frac{\rho}{\rho_o}\sigma_1 - \frac{1}{\rho_o e_1}\frac{\partial p_s}{\partial i} + D_u^{\mathrm{lU}} + \frac{1}{e_3}\frac{\partial}{\partial k}\left(\frac{A^{vm}}{e_3}\frac{\partial u}{\partial k}\right)$$

$$\zeta = \frac{1}{e_1 e_2}\left(\frac{\partial}{\partial i}\left[e_2 v\right] - \frac{\partial}{\partial j}\left[e_1 u\right]\right)$$

$$\chi = \frac{1}{e_1 e_2 e_3}\left(\frac{\partial}{\partial i}\left[e_2 e_3\, u\right] + \frac{\partial}{\partial j}\left[e_1 e_3\, v\right]\right)$$

with

$$\frac{\partial v}{\partial t} = -(\zeta + f)\, u - \frac{1}{e_3}\,\omega\,\frac{\partial v}{\partial k} - \frac{1}{2e_2}\frac{\partial}{\partial j}\left(u^2 + v^2\right)$$
$$-\frac{1}{\rho_o e_2}\frac{\partial p_h}{\partial j} + g\frac{\rho}{\rho_o}\sigma_2 - \frac{1}{\rho_o e_2}\frac{\partial p_s}{\partial j} + D_v^{\mathrm{lU}} + \frac{1}{e_3}\frac{\partial}{\partial k}\left(\frac{A^{vm}}{e_3}\frac{\partial v}{\partial k}\right)$$

$$\overline{M_u} = \frac{1}{H}\int_{-H}^{0}\left[\frac{\partial u}{\partial t} + \frac{1}{\rho_o e_1}\frac{\partial p_s}{\partial i}\right] e_3\, dk$$

$$\overline{M_v} = \frac{1}{H}\int_{-H}^{0}\left[\frac{\partial v}{\partial t} + \frac{1}{\rho_o e_2}\frac{\partial p_s}{\partial j}\right] e_3\, dk$$

$$\frac{\partial p_h}{\partial k} = -\rho\, g\, e_3$$

$$\sigma_1 = \frac{1}{e_1}\frac{\partial z}{\partial i}\bigg|_s \quad , \quad \text{and} \quad \sigma_2 = \frac{1}{e_2}\frac{\partial z}{\partial j}\bigg|_s$$

$$\frac{\partial \omega}{\partial k} = -e_3\,\chi$$

$$\frac{1}{\rho_o e_1}\frac{\partial p_s}{\partial i} = \overline{M_u} + \frac{1}{H e_2}\frac{\partial}{\partial j}\left(\frac{\partial \psi}{\partial t}\right)$$

$$\frac{1}{\rho_o e_2}\frac{\partial p_s}{\partial j} = \overline{M_v} - \frac{1}{H e_1}\frac{\partial}{\partial i}\left(\frac{\partial \psi}{\partial t}\right)$$

with $$\frac{\partial}{\partial i}\left[\frac{e_2}{H e_1}\frac{\partial}{\partial i}\left(\frac{\partial \psi}{\partial t}\right)\right] + \frac{\partial}{\partial j}\left[\frac{e_1}{H e_2}\frac{\partial}{\partial j}\left(\frac{\partial \psi}{\partial t}\right)\right] = \frac{\partial}{\partial i}\left(e_2\overline{M_v}\right) - \frac{\partial}{\partial j}\left(e_1\overline{M_u}\right)$$

$$\frac{\partial T}{\partial t} = -\frac{1}{e_1 e_2 e_3}\left[\frac{\partial}{\partial i}\left(e_2 e_3\, T\, u\right) + \frac{\partial}{\partial j}\left(e_1 e_3\, T\, v\right)\right]$$
$$-\frac{1}{e_3}\frac{\partial}{\partial k}\left(T\,\omega\right) + D^{lT} + \frac{1}{e_3}\frac{\partial}{\partial k}\left(\frac{A^{vT}}{e_3}\frac{\partial T}{\partial k}\right)$$

and

$$\frac{\partial S}{\partial t} = -\frac{1}{e_1 e_2 e_3}\left[\frac{\partial}{\partial i}\left(e_2 e_3\, S\, u\right) + \frac{\partial}{\partial j}\left(e_1 e_3\, S\, v\right)\right]$$
$$-\frac{1}{e_3}\frac{\partial}{\partial k}\left(S\,\omega\right) + D^{lS} + \frac{1}{e_3}\frac{\partial}{\partial k}\left(\frac{A^{vT}}{e_3}\frac{\partial S}{\partial k}\right)$$

where the expression of $\left(D_u^{\mathrm{lU}}\ ,\ D_v^{\mathrm{lU}}\right)$ and $\left(D^{lT}\ ,\ D^{lS}\right)$ is given in Table I.3 and I.4, respectively.

Table I.2: Set of equations solved by the model in the curvilinear s-coordinate system.

I.4-b The *s*-Coordinate Formulation

Starting from the set of equations established in I.3 for the special case $k = z$ and thus $e_3 = 1$, we introduce an arbitrary vertical coordinate $s = s(i, j, z)$, which includes *z*- and σ-coordinates as special cases ($s = z$ and $s = \sigma = z/H$, resp.). A formal derivation of the transformed equations is given in Appendix A. Let us define the vertical scale factor by $e_3 = \partial z/\partial s$ (e_3 is now a function of (i, j, k)), and the slopes in the $(\mathbf{i}, \mathbf{j})$ directions between *s*- and *z*-surfaces by :

$$\sigma_1 = \frac{1}{e_1}\frac{\partial z}{\partial i}\bigg|_s \quad , \quad \text{and} \quad \sigma_2 = \frac{1}{e_2}\frac{\partial z}{\partial j}\bigg|_s \qquad \text{(I.4.2)}$$

We also introduce a "vertical" velocity ω defined as the velocity normal to *s*-surfaces:

$$\omega = w - \sigma_1 u - \sigma_2 v \qquad \text{(I.4.3)}$$

The equations solved by the ocean model in the rigid-lid approximation (i.e. Eqs. (I.1.1) to (I.1.6) plus Eqs. (I.2.3) and (I.1.4)) in *s*-coordinates can be written as follows:

* momentum equation:

$$\frac{\partial u}{\partial t} = +(\zeta + f)\, v - \frac{1}{e_3}\,\omega\,\frac{\partial u}{\partial k}$$
$$-\frac{1}{e_1}\frac{\partial}{\partial i}\left(\frac{1}{2}\left(u^2 + v^2\right) + \frac{p_h}{\rho_o}\right) \qquad \text{(I.4.4)}$$
$$+ g\frac{\rho}{\rho_o}\sigma_1 - \frac{1}{\rho_o e_1}\frac{\partial p_s}{\partial i} + D_u^{\mathrm{U}}$$

$$\frac{\partial v}{\partial t} = -(\zeta + f)u - \frac{1}{e_3}\omega\frac{\partial v}{\partial k} - \frac{1}{e_2}\frac{\partial}{\partial j}\left(\frac{1}{2}(u^2+v^2) + \frac{p_h}{\rho_o}\right) + g\frac{\rho}{\rho_o}\sigma_2 - \frac{1}{\rho_o e_2}\frac{\partial p_s}{\partial j} + D_v^{\mathbf{U}} \tag{1.4.5}$$

where the relative vorticity, ζ, the surface pressure gradient, and the hydrostatic pressure have the same expressions as in z-coordinates although they do not represent exactly the same quantities. ω is provided by the same equation as w, i.e. (I.3.14), with χ, the divergence of the horizontal velocity field given by :

$$\chi = \frac{1}{e_1 e_2 e_3}\left[\frac{\partial(e_2 e_3 u)}{\partial i} + \frac{\partial(e_1 e_3 v)}{\partial j}\right] \tag{I.4.6}$$

* tracer equations:

$$\frac{\partial T}{\partial t} = -\frac{1}{e_1 e_2 e_3}\left[\frac{\partial(e_2 e_3 T u)}{\partial i} + \frac{\partial(e_1 e_3 T v)}{\partial j}\right] - \frac{1}{e_3}\frac{\partial(T\omega)}{\partial k} + D^T \tag{I.4.7}$$

$$\frac{\partial S}{\partial t} = -\frac{1}{e_1 e_2 e_3}\left[\frac{\partial(e_2 e_3 S u)}{\partial i} + \frac{\partial(e_1 e_3 S v)}{\partial j}\right] - \frac{1}{e_3}\frac{\partial(S\omega)}{\partial k} + D^S \tag{I.4.8}$$

The equation of state have the same expression as in z-coordinates. The expression of $\mathbf{D}^{\mathbf{U}}$, D^S and D^T depends on the subgrid scale parameterization used. It will be defined in § I.5. The whole set of the continuous equations solved by the model in the s-coordinate systeme is summmerized in Table I.2.

References

(see OPA-Bibliography when the year is in **bold**)

Beckmann, A., and D. Haidvoguel, 1993: Numerical simulation of flow around a tall isolated seamont. *J. Phys. Oceanogr, 23*, 1736-1753.

Gerdes, R., 1993a: A primitive equation ocean circulation model using a general vertical coordinate transformation. Part 1: description and testing of the model. *J. Geophys. Res., 98*, C8, 14,683-14,701.

———, 1993b: A primitive equation ocean circulation model using a general vertical coordinate transformation. Part 2: application to an overflow problem. *J. Geophys. Res., 98*, C8, 14,703-14,726.

Haney, R. L., 1991: On the pressure gradient force over steep topography in sigma coordinate ocean models. *J. Phys. Oceanogr., 21*, 610-619.

I.5 SUBGRID SCALE PHYSICS

The primitive equations describe the behaviour of a geophysical fluid at space and time scales larger than a few kilometers in the horizontal, a few meters in the vertical and a few minutes. They are usually solved at larger scales, the specified grid spacing and time step of the numerical model. The effects of smaller scale motions (coming from the advective terms in the Navier-Stokes equations) must be represented entirely in terms of large scale patterns to close the equations. These effects appear in the equations as the divergence of turbulent fluxes (i.e. fluxes associated with the mean correlation of small scale perturbations). Assuming a turbulent closure hypothesis is equivalent to chose a formulation for these fluxes. It is usually called the subgrid scale physics. It must be emphasized that this is the weakest part of the primitive equations, but also one of the most important for long term simulations as small scale processes *in fine* balance the surface input of kinetic energy and heat.

The control exerted by gravity on the flow induces a strong anisotropy between the lateral and vertical motions. Therefore subgrid-scale physics $\mathbf{D}^{\mathbf{U}}$, D^T and D^S in (I.1.1), (I.1.4) and (I.1.5) are divided into a lateral part $\mathbf{D}^{l\mathbf{U}}$, D^{lT}, D^{lS} and a vertical part $\mathbf{D}^{v\mathbf{U}}$, D^{vT}, D^{vS}. The formulation of these terms and their underlying physics are briefly discussed in the next two sub-sections.

I.5-a Vertical Subgrid Scale Physics

The model resolution is always larger than the scale at which the major sources of vertical turbulence occurs (shear instability, internal wave breaking, ...). Turbulent motions are thus never explicitly solved, even partially, but always parameterized. The vertical turbulent fluxes are assumed to depend linearly on the gradients of large-scale quantities (for example, the turbulent heat flux is given by $\overline{T'w'} = -A^{vT}\partial_z \overline{T}$, where A^{vT} is an eddy coefficient). This formulation is analogous to that of molecular diffusion and dissipation. This is quite clearly a necessary compromise: considering only the molecular viscosity acting on large scale severely underestimates the role of turbulent diffusion and dissipation, while an accurate consideration of the details of turbulent motions is

simply impractical. The resulting vertical momentum and tracer diffusive operators are of second order :

$$\mathbf{D}^{vU} = \frac{\partial}{\partial z}\left(A^{vm} \frac{\partial \mathbf{U}_h}{\partial z} \right), \quad D^{vT} = \frac{\partial}{\partial z}\left(A^{vT} \frac{\partial T}{\partial z} \right), \quad D^{vS} = \frac{\partial}{\partial z}\left(A^{vS} \frac{\partial S}{\partial z} \right) \tag{I.5.1}$$

where A^{vm} and A^{vT} are the vertical eddy viscosity and diffusivity coefficients, respectively. At the sea surface and at the bottom, turbulent fluxes of momentum, heat and salt must be specified (see § III.4 and III.9). All the vertical physics is embedded in the specification of the eddy coefficients. They can be assumed to be either constant, or function of the local fluid properties (as Richardson number, Brunt-Vaisälä frequency, ...), or computed from a turbulent closure model. The choices available in OPA are discussed in § III.7.

I.5-b Lateral Diffusive and Viscous Operators

Lateral turbulence can be roughly divided into a mesoscale turbulence associated to eddies which can be solved explicitly if the resolution is sufficient as their underlying physics are included in the primitive equations, and a sub mesoscale turbulence which is never explicitly solved even partially, but always parameterized. The formulation of lateral eddy fluxes depends on whether the mesoscale is below or above the gridspacing (i.e. the model is eddy-resolving or not).

In non-eddy resolving configurations, the closure is similar to that used for the vertical physics. The lateral turbulent fluxes are assumed to depend linearly on the lateral gradients of large-scale quantities. The resulting lateral diffusive and dissipative operators are of second order. Observations show that lateral mixing induced by mesoscale turbulence tends to be along isopycnal surfaces (or more precisely neutral surfaces, i.e. isopycnal surfaces referenced at the local depth) rather than across them. As the slope of isopycnal surfaces is small in the ocean, a common approximation is to assume that the 'lateral' direction is the horizontal, i.e. the lateral mixing is performed along geopotential surfaces. This leads to a geopotential second order operator for lateral subgrid scale physics. This assumption can be relaxed: the eddy-induced turbulent fluxes can be better approached by assuming that they depend linearly on the gradients of large-scale quantities computed along isopycnal surfaces. In such a case, the diffusive operator is an isopycnal second order operator and it has components in the three space directions. However, both horizontal and isopycnal operators have no effect on mean (i.e. large scale) potential energy whereas potential energy is a main source of turbulence (through baroclinic instabilities). Gent and McWilliams [1990] have proposed a parameterization of mesoscale eddy-induced turbulence which associates an eddy-induced velocity to the isopycnal diffusion. Its mean effect is to reduce the mean potential energy of the ocean. This leads to a formulation of lateral subgrid scale physics made up of an isopycnal second order operator and an eddy induced advective part. In all these lateral diffusive formulations, the specification of the lateral eddy coefficients remains the problematic point as there is no satisfactory formulation of these coefficients as a function of large scale features.

In eddy-resolving configurations, a second order operator can be used, but usually a more scale selective one (biharmonic operator) is preferred as the gridspacing is usually not small enough compared to the scale of the eddies. The role devoted to the subgrid scale physics is to dissipate the energy that cascades toward the grid scale and thus ensures the stability of the model while not interfering with the solved mesoscale activity.

All these parameterizations of subgrid scale physics present advantages and disadvantages. There are not all available in OPA. In the *z*-coordinate formulation, four options are offered for active tracers (temperature and salinity): second order geopotential operator, second order isopycnal operator, Gent and McWilliams [1990] parameterization and fourth order geopotential operator. The same options are available for momentum, except Gent and McWilliams [1990] parameterization which only involves tracers. In *s*-coordinate formulation, an additional option is offered for tracers: second order operator acting along *s*-surfaces, and for momentum: fourth order operator acting along *s*-surfaces (see §III.6).

* lateral second order tracer diffusive operator :

The lateral second order tracer diffusive operator is defined by (see Appendix B):

$$D^{lT} = \nabla \cdot \left(A^{lT} \Re \nabla T \right) \quad \text{with} \quad \Re = \begin{pmatrix} 1 & 0 & -r_1 \\ 0 & 1 & -r_2 \\ -r_1 & -r_2 & r_1^2 + r_2^2 \end{pmatrix} \tag{I.5.2}$$

where r_1 and r_2 are the slopes between the surface along which the diffusive operator acts and the surface of computation (*z*- or *s*-surfaces), and ∇ is the differential operator defined in § I.3 or § I.4 depending on the vertical coordinate used (see Table I.3). Note that the formulation of $\Re$ is exact for the slopes between geopotential and *s*-surfaces, while it is only an approximation for the slopes between isopycnal and *z*- or *s*-surfaces. Indeed, in the latter case, two assumptions are made to simplify $\Re$ [Cox, 1987]: the ratio between lateral and vertical diffusive coefficients is known to be several orders of magnitude smaller than unity, and the slopes are, generally less than 10^{-2} in the ocean (see Appendix B). This leads to the linear tensor (I.5.2) where the two isopycnal directions of diffusion are independent and where the diapycnal diffusivity contribution is solely along the vertical.

second order lateral diffusive operator on tracers:

* *z*-coordinates:

$$D^{lT} = \frac{1}{e_1 e_2}\left\{ \frac{\partial}{\partial i}\left[A^{lT}\left(\frac{e_2}{e_1}\frac{\partial T}{\partial i} - r_1 \frac{e_2}{e_3}\frac{\partial T}{\partial k} \right)\right] + \frac{\partial}{\partial j}\left[A^{lT}\left(\frac{e_1}{e_2}\frac{\partial T}{\partial j} - r_2 \frac{e_1}{e_3}\frac{\partial T}{\partial k} \right)\right]\right\} + \frac{1}{e_3}\frac{\partial}{\partial k}\left[A^{lT}\left(-\frac{r_1}{e_1}\frac{\partial T}{\partial i} - \frac{r_2}{e_2}\frac{\partial T}{\partial j} + \frac{\left(r_1^2 + r_2^2\right)}{e_3}\frac{\partial T}{\partial k} \right)\right]$$

where $r_1 = r_2 = 0$ for geopotential diffusion and $r_1 = \frac{e_3}{e_1}\left(\frac{\partial \rho}{\partial i}\right)\left(\frac{\partial \rho}{\partial k}\right)^{-1}$, $r_2 = \frac{e_3}{e_2}\left(\frac{\partial \rho}{\partial j}\right)\left(\frac{\partial \rho}{\partial k}\right)^{-1}$ for isopycnal diffusion

**s*-coordinates:

$$D^{lT} = \frac{1}{e_1 e_2 e_3}\left[\frac{\partial}{\partial i}\left[A^{lT}\left(\frac{e_2 e_3}{e_1}\frac{\partial T}{\partial i}\bigg|_s - e_2 r_1 \frac{\partial T}{\partial s} \right)\right]\bigg|_s + \frac{\partial}{\partial j}\left[A^{lT}\left(\frac{e_1 e_3}{e_2}\frac{\partial T}{\partial j}\bigg|_s - e_1 r_2 \frac{\partial T}{\partial s} \right)\right]\bigg|_s + \frac{\partial}{\partial s}\left[A^{lT}\left(-e_2 r_1 \frac{\partial T}{\partial i}\bigg|_s - e_1 r_2 \frac{\partial T}{\partial j}\bigg|_s + \frac{e_1 e_2}{e_3}\left(r_1^2 + r_2^2\right)\frac{\partial T}{\partial s} \right)\right] \right]$$

where $r_1 = \frac{1}{e_1}\frac{\partial z}{\partial i}\Big|_s$, $r_2 = \frac{1}{e_2}\frac{\partial z}{\partial j}\Big|_s$ for geopotential diffusion

and $r_1 = \frac{1}{e_1}\frac{\partial z}{\partial i}\Big|_s + \frac{e_3}{e_1}\left(\frac{\partial \rho}{\partial i}\Big|_s\right)\left(\frac{\partial \rho}{\partial s}\right)^{-1}$, $r_2 = \frac{1}{e_2}\frac{\partial z}{\partial j}\Big|_s + \frac{e_3}{e_2}\left(\frac{\partial \rho}{\partial j}\Big|_s\right)\left(\frac{\partial \rho}{\partial s}\right)^{-1}$ for isopycnal diffusion

fourth order tracer diffusive operator: (diffusion along geopotential or *s*-surfaces only)

$$D^{lT} = \Delta\left(A^{lT}\ \Delta\ T\right)$$

where Δ is the second order lateral diffusive operator defined above in *z*- or *s*-coordinates

Table I.3: tracer diffusive operators used to represent lateral subgrid scale processes in the curvilinear z- and s-coordinate system.

For *geopotential* diffusion, r_1 and r_2 are the slopes between the geopotential and computational surfaces: in *z*-coordinates they are zero ($r_1 = r_2 = 0$) while in *s*-coordinate they are equal to σ_1 and σ_2, respectively (see (I.4.2)).

For *isopycnal* diffusion, r_1 and r_2 are the slopes between the isopycnal and geopotential surfaces.
In *z*-coordinates they are given by:

$$r_1 = \frac{e_3}{e_1}\left(\frac{\partial \rho}{\partial i}\right)\left(\frac{\partial \rho}{\partial k}\right)^{-1}, \quad r_2 = \frac{e_3}{e_2}\left(\frac{\partial \rho}{\partial j}\right)\left(\frac{\partial \rho}{\partial k}\right)^{-1} \tag{I.5.3}$$

while in *s*-coordinate they are given by

$$r_1 = \sigma_1 + \frac{e_3}{e_1}\left(\frac{\partial \rho}{\partial i}\Big|_s\right)\left(\frac{\partial \rho}{\partial s}\right)^{-1}$$
$$r_2 = \sigma_2 + \frac{e_3}{e_2}\left(\frac{\partial \rho}{\partial j}\Big|_s\right)\left(\frac{\partial \rho}{\partial s}\right)^{-1} \tag{I.5.4}$$

For Gent and McWilliams [1990] diffusion, an additional tracer advection is used in combination with the isopycnal diffusion of tracers:

$$D^{lT} = \nabla.\left(A^{lT}\ \Re\ \nabla T\right) + \nabla.\left(\mathbf{U}^{*}\ T\right) \tag{I.5.5}$$

where $\mathbf{U}^{*} = \left(u^{*}, v^{*}, w^{*}\right)$ is a non-divergent, eddy-induced transport velocity. This velocity field is defined from r_1

second order diffusive operator on momentum:

* geopotential diffusion (*z*-coordinates) or diffusion along *s*-surfaces (*s*-coordinates):

$$\mathbf{D}^{lU} = \nabla_h\left(A^{lm}\chi\right) - \nabla_h \times \left(A^{lm}\,\zeta\,\mathbf{k}\right) = \begin{pmatrix} \dfrac{1}{e_1}\dfrac{\partial}{\partial i}\left[A^{lm}\chi\right] - \dfrac{1}{e_2 e_3}\dfrac{\partial}{\partial j}\left[A^{lm}\,e_3\zeta\right] \\ \dfrac{1}{e_2}\dfrac{\partial}{\partial j}\left[A^{lm}\chi\right] + \dfrac{1}{e_1 e_3}\dfrac{\partial}{\partial i}\left[A^{lm}\,e_3\zeta\right] \end{pmatrix}$$

*geopotential diffusion (*s*-coordinates) or isopycnal diffusion (*z*- and *s*-coordinates):

$$\mathbf{D}^{lU} = \nabla.\left(A^{lm}\,\Re\,\nabla\mathbf{U}_h\right) = \frac{1}{e_1 e_2 e_3}\left[\frac{\partial}{\partial i}\left[A^{lm}\left(\frac{e_2 e_3}{e_1}\frac{\partial \mathbf{U}_h}{\partial i}\bigg|_s - e_2 r_1 \frac{\partial \mathbf{U}_h}{\partial s}\right)\right]_s + \frac{\partial}{\partial j}\left[A^{lm}\left(\frac{e_1 e_3}{e_2}\frac{\partial \mathbf{U}_h}{\partial j}\bigg|_s - e_1 r_2 \frac{\partial \mathbf{U}_h}{\partial s}\right)\right]_s \right.$$
$$\left. + \frac{\partial}{\partial s}\left[A^{lm}\left(-e_2 r_1 \frac{\partial T}{\partial i}\bigg|_s - e_1 r_2 \frac{\partial T}{\partial j}\bigg|_s + \frac{e_1 e_2}{e_3}\left(r_1^2 + r_2^2\right)\frac{\partial T}{\partial s}\right)\right] \right]$$

where $r_1 = \dfrac{1}{e_1}\dfrac{\partial z}{\partial i}\bigg|_s$, $r_2 = \dfrac{1}{e_2}\dfrac{\partial z}{\partial j}\bigg|_s$ for geopotential diffusion in *s*-coordinates,

$r_1 = \dfrac{e_3}{e_1}\left(\dfrac{\partial \rho}{\partial i}\bigg|_s\right)\left(\dfrac{\partial \rho}{\partial s}\right)^{-1}$, $r_2 = \dfrac{e_3}{e_2}\left(\dfrac{\partial \rho}{\partial j}\bigg|_s\right)\left(\dfrac{\partial \rho}{\partial s}\right)^{-1}$ for isopycnal diffusion in *z*-coordinates

$r_1 = \dfrac{1}{e_1}\dfrac{\partial z}{\partial i}\bigg|_s + \dfrac{e_3}{e_1}\left(\dfrac{\partial \rho}{\partial i}\bigg|_s\right)\left(\dfrac{\partial \rho}{\partial s}\right)^{-1}$, $r_2 = \dfrac{1}{e_2}\dfrac{\partial z}{\partial j}\bigg|_s + \dfrac{e_3}{e_2}\left(\dfrac{\partial \rho}{\partial j}\bigg|_s\right)\left(\dfrac{\partial \rho}{\partial s}\right)^{-1}$ for isopycnal diffusion in s-coordinates.

<u>fourth order diffusive operator on momentum:</u>

$\mathbf{D}^{lU} = \Delta\left(A^{lU}\,\Delta\,\mathbf{U}_h\right)$ where Δ is defined by $\Delta\begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} \dfrac{1}{e_1}\dfrac{\partial}{\partial i}\left(\dfrac{1}{e_1 e_2}\left[\dfrac{\partial(e_2 a)}{\partial i} + \dfrac{\partial(e_1 b)}{\partial j}\right]\right) - \dfrac{1}{e_2 e_3}\dfrac{\partial}{\partial j}\left(\dfrac{e_3}{e_1 e_2}\left[\dfrac{\partial(e_2 b)}{\partial i} - \dfrac{\partial(e_1 a)}{\partial j}\right]\right) \\ \dfrac{1}{e_2}\dfrac{\partial}{\partial j}\left(\dfrac{1}{e_1 e_2}\left[\dfrac{\partial(e_2 a)}{\partial i} + \dfrac{\partial(e_1 b)}{\partial j}\right]\right) + \dfrac{1}{e_1 e_3}\dfrac{\partial}{\partial i}\left(\dfrac{e_3}{e_1 e_2}\left[\dfrac{\partial(e_2 b)}{\partial i} - \dfrac{\partial(e_1 a)}{\partial j}\right]\right) \end{pmatrix}$

the 2nd order diffusive operator along *s*-surfaces in *s*-coordinates or along geopotential surfaces in *z*-coordinates,

and by $\Delta\begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} \nabla.\left(\Re\,\nabla a\right) \\ \nabla.\left(\Re\,\nabla b\right) \end{pmatrix}$ with $\nabla.\left(\Re\,\nabla\bullet\right)$, the second order geopotential diffusive operator, in *s*-coordinates

Table I.4: momentum diffusive operators used to represent lateral subgrid scale processes in the curvilinear z- and s-coordinate system.

and r_2, the isopycnal slopes evaluated as in (I.5.3) or (I.5.4) depending on the vertical coordinate used, and A^{eiv} an eddy induced velocity coefficient (or equivalently the isopycnal thickness diffusivity coefficient). It takes the following expression:

$$u^* = \frac{1}{e_3}\frac{\partial}{\partial k}\left[A^{eiv}\,r_1\right]$$
$$v^* = \frac{1}{e_3}\frac{\partial}{\partial k}\left[A^{eiv}\,r_2\right] \qquad \text{(I.5.6)}$$
$$w^* = -\frac{1}{e_1 e_2}\left[\frac{\partial}{\partial i}\left(A^{eiv}\,e_2 r_1\right) + \frac{\partial}{\partial j}\left(A^{eiv}\,e_1 r_2\right)\right]$$

The normal component of the eddy induced velocity is zero at all the boundaries by tapering either the eddy coefficient or the slopes to zero in the vicinity of the boundaries.

* lateral fourth order tracer diffusive operator:

The lateral fourth order tracer diffusive operator is defined by:

$$D^{lT} = \Delta\left(A^{lT}\,\Delta T\right) \quad \text{where } \Delta(\bullet) = \nabla.\left(\Re\,\nabla\bullet\right) \qquad \text{(I.5.7)}$$

It is the second order operator given by (I.5.2) applied twice with the eddy diffusion coefficient correctly placed.

* lateral second order momentum diffusive operator

The second order momentum diffusive operator along *z*- or *s*-surfaces is found by applying (I.3.4) to the horizontal velocity vector (see Appendix B):

$$\mathbf{D}^{lU} = \nabla_h \left(A^{lm} \chi \right) - \nabla_h \times \left(A^{lm}\, \zeta\, \mathbf{k} \right)$$

$$= \begin{pmatrix} \dfrac{1}{e_1} \dfrac{\partial \left(A^{lm} \chi \right)}{\partial i} - \dfrac{1}{e_2 e_3} \dfrac{\partial \left(A^{lm}\, e_3 \zeta \right)}{\partial j} \\ \dfrac{1}{e_2} \dfrac{\partial \left(A^{lm} \chi \right)}{\partial j} + \dfrac{1}{e_1 e_3} \dfrac{\partial \left(A^{lm}\, e_3 \zeta \right)}{\partial i} \end{pmatrix} \tag{I.5.8}$$

Such a formulation ensure a complete separation between the vorticity and horizontal divergence fields (§ II.4-c). Unfortunately, it is not available for geopotential diffusion in *s*-coordinates and for isopycnal diffusion. In these two cases, the *u*- and *v*-fields are considered as independent scalar fields, so that the diffusive operator is given by:

$$\begin{aligned} D_u^{lU} &= \nabla . \left(\Re\, \nabla u \right) \\ D_v^{lU} &= \nabla . \left(\Re\, \nabla v \right) \end{aligned} \tag{I.5.9}$$

where $\Re$ is given by (I.5.2). It is the same expression as those used for diffusive operator on tracers.

* lateral fourth order momentum diffusive operator

As for tracers, the fourth order momentum diffusive operator along *z*- or *s*-surfaces is a re-entering second order operator (I.5.8) or (I.5.9) with the eddy viscosity coefficient correctly placed:

geopotential diffusion in *z*-coordinates:

$$\begin{aligned} \mathbf{D}^{lU} = \nabla_h \Big\{ & \nabla_h \cdot \left[A^{lm} \nabla_h \left(\chi \right) \right] \Big\} \\ & + \nabla_h \times \Big\{ \mathbf{k} \cdot \nabla \times \left[A^{lm} \nabla_h \times \left(\zeta\, \mathbf{k} \right) \right] \Big\} \end{aligned} \tag{I.5.10}$$

geopotential diffusion in *s*-coordinates:

$$\begin{aligned} D_u^{lU} &= \Delta . \left(A^{lm}\, \Delta u \right) \\ D_v^{lU} &= \Delta . \left(A^{lm}\, \Delta v \right) \end{aligned} \quad \text{where } \Delta(\bullet) = \nabla . \left(\Re\, \nabla \bullet \right) \tag{I.5.11}$$

The whole set of the continuous equations solved by the model in the *z*-coordinate systeme is summmerized in Table I.3.

References

(see OPA-Bibliography when the year is in **bold**)

Gent, P. R., and J. C. McWilliams, 1990 : Isopycnal mixing in ocean circulation models. *J. Phys. Oceanogr., Notes and Correspondence, 20*, 150-155.

Cox, M., 1987 : Isopycnal diffusion in a z-coordinate ocean model. *Ocean Modelling*, 74, 1-9.

II. DISCRETIZATION

II.1 INTRODUCTION

II.1-a Arrangement of Variables

The numerical techniques used to solve the Primitive Equations are based on the traditional, centered second-order finite difference approximation. Special attention has been given to the homogeneity of the solution in the three space directions. The arrangement of variables is the same in all directions (Fig. II.1). It consists in cells centered on scalar points (T, S, p, ρ, χ) with vector points (u, v, w) defined in the centre of each face of the cells. This is the generalization to three dimension of the well-known "C" grid in Arakawa's classification. The relative and planetary vorticity, ζ and f, are defined in the center of each vertical edge and the barotropic stream function ψ is defined at horizontal points overlying the ζ and f-points.

The ocean mesh (i.e. the position of all the scalar and vector points) is defined by the transformation that gives (λ, φ, z) as a function of (i, j, k). The grid-points are located at integer or integer and a half values of (i, j, k) as indicated on table II.1. In all the following, subscripts u, v, w, f, uw, vw or fw indicate the position of the grid-point where the scale factors are defined. Each scale factor is defined as the local analytical value provided by (I.3.1). As a result, the mesh on which partial derivatives $\partial/\partial i$, $\partial/\partial j$, and $\partial/\partial k$ are evaluated is uniform mesh with a grid size unity. Discrete partial derivative are formulated by the traditional, centered second-order finite difference approximation while the scale factors are chosen equal to their local analytical value. An important point here is that the partial derivative of the scale factors must be evaluated by centered second-order finite difference approximation, not from their analytical expression. This preserves the symmetry of the discrete set of equations and therefore allows to satisfy many of the continuous properties (see Annexe C).

II.1-b Discrete Operators

Given the values of a variable q at adjacent points, the derivation and averaging operators at the midpoint between them are:

$$\delta_i[q] = q(i+1/2) - q(i-1/2) \qquad \text{(II.1.1)}$$

$$\overline{q}^{\,i} = \{q(i+1/2) + q(i-1/2)\}/2 \qquad \text{(II.1.2)}$$

Similar operator are defined with respect to $i+1/2$, j, $j+1/2$, k, and $k+1/2$. Following (I.3.2) and (I.3.5), the gradient of a variable q defined at T-point has its three components defined at (u,v,w) while its laplacian is

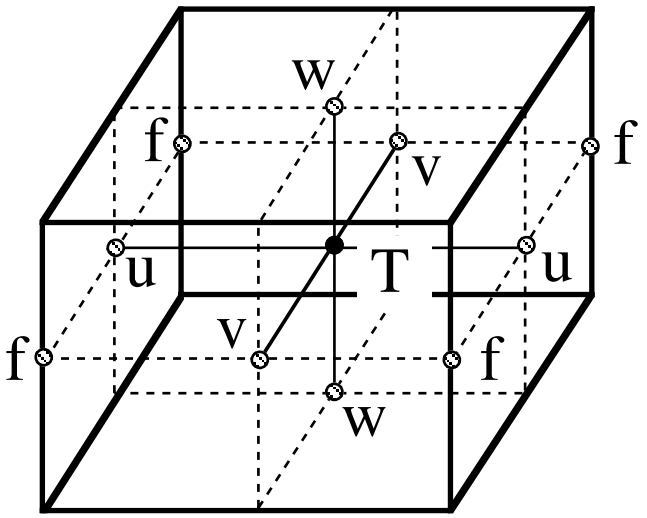


Figure II.1 : Arrangement of variables. T indicates scalar points where temperature, salinity, density, pressure and horizontal divergence are defined. (u,v,w) indicates vector points, and f indicates vorticity points where both relative and planetary vorticities are defined.

T	i	j	k
u	$i+1/2$	j	k
v	i	$j+1/2$	k
w	i	j	$k+1/2$
f	$i+1/2$	$j+1/2$	k
uw	$i+1/2$	j	$k+1/2$
vw	i	$j+1/2$	$k+1/2$
fw	$i+1/2$	$j+1/2$	$k+1/2$

Table II.1 : Location of grid-points as a function of integer or integer and a half value of the column, line or level. Note that in the FORTRAN code, the vertical indexation is re-oriented downward (see §III.1).

defined at *T*-point. These operators have the following discrete forms in the curvilinear *s*-coordinate system :

$$\nabla q \equiv \frac{1}{e_{1u}} \delta_{i+1/2}[q]\,\mathbf{i} + \frac{1}{e_{2v}} \delta_{j+1/2}[q]\,\mathbf{j} + \frac{1}{e_{3w}} \delta_{k+1/2}[q]\,\mathbf{k} \quad \text{(II.1.3)}$$

$$\Delta q \equiv \frac{1}{e_{1T}\, e_{2T}\, e_{3T}} \left(\delta_i \left[\frac{e_{2u} e_{3u}}{e_{1u}} \delta_{i+1/2}[q] \right] + \delta_j \left[\frac{e_{1v} e_{3v}}{e_{2v}} \delta_{j+1/2}[q] \right] \right) + \frac{1}{e_{3T}} \delta_k \left[\frac{1}{e_{3w}} \delta_{k+1/2}[q] \right] \quad \text{(II.1.4)}$$

Following (I.3.3) and (I.3.4), a vector $\mathbf{A} = (a_1, a_2, a_3)$ defined at vector points (u,v,w) has its three curl components defined at (vw,uw,f) and its divergence defined at *T*-points:

$$\nabla \times \mathbf{A} \equiv \frac{1}{e_{2v}\, e_{3vw}} \left(\delta_{j+1/2}[e_{3w} a_3] - \delta_{k+1/2}[e_{2v} a_2] \right) \mathbf{i} + \frac{1}{e_{2u}\, e_{3uw}} \left(\delta_{k+1/2}[e_{1u} a_1] - \delta_{i+1/2}[e_{3w} a_3] \right) \mathbf{j} + \frac{1}{e_{1f}\, e_{2f}} \left(\delta_{i+1/2}[e_{2v} a_2] - \delta_{j+1/2}[e_{1u} a_1] \right) \mathbf{k} \quad \text{(II.1.5)}$$

$$\nabla \cdot \mathbf{A} \equiv \frac{1}{e_{1T}\, e_{2T}\, e_{3T}} \left(\delta_i [e_{2u} e_{3u}\, a_1] + \delta_j [e_{1v} e_{3v}\, a_2] \right) + \frac{1}{e_{3T}} \delta_k [a_3] \quad \text{(II.1.6)}$$

(II.1.3) and (II.1.5) have exactly the same expression in the curvilinear *z*-coordinates system, while (II.1.4) and (II.1.6) can be simplified in such a case: the vertical scale factor is a function of the single variable *k* and does not depend on the horizontal location of a grid point so that it can be simplified from outside and inside the δ_i and δ_j operators. The vertical average over the whole water column denoted by an overbar becomes for a quantity *q* which is a masked field (i.e. equal to zero inside solid area):

$$\bar{q} = \frac{1}{H} \int_{k^b}^{k^o} q\, e_{3q}\, dk \equiv \frac{1}{H_q} \sum_k q\, e_{3q} \quad \text{(II.1.7)}$$

where H_q is the masked sum of the vertical scale factors at q points, k^b and k^o are the bottom and surface *k*-index, and the symbol $\sum$ referring to a summation over all grid points of the same species in the direction indicated by the subscript (here *k*). In continuous, the following properties are satisfied :

$$\nabla \times \nabla q = \vec{0} \quad \text{(II.1.8)}$$

$$\nabla \cdot (\nabla \times \mathbf{A}) = 0 \quad \text{(II.1.9)}$$

It is straightforward to demonstrate that these properties are verified locally in discrete form as soon as the scalar *q* is taken at *T*-points and the vector **A** has its components defined at vector points (*u,v,w*).

Let *a* and *b* be two fields defined on the ocean mesh, extended to zero inside continental area. By an integration by part it is obvious to demonstrate that the derivation operators (δ_i, δ_j, and δ_k) are anti-symmetric linear operators, and further that the averaging operators ($\overline{\cdot}^i$, $\overline{\cdot}^j$, and $\overline{\cdot}^k$) are symmetric linear operators, i.e.,

$$\sum_i a_i\, \delta_i[b] \equiv -\sum_i \delta_{i+1/2}[a]\, b_{i+1/2} \quad \text{(II.1.10)}$$

$$\sum_i a_i\, \overline{b}^i \equiv \sum_i \overline{a}^{i+1/2}\, b_{i+1/2} \quad \text{(II.1.11)}$$

In other words, the adjoint of averaging and derivation operators are $\overline{\cdot}^{i\,*} = \overline{\cdot}^{i+1/2}$ and $\delta_i^{\,*} = -\delta_{i+1/2}$, respectively. This two properties will be used extensively in § II.4 and in Appendix C to demonstrate integral conservative properties of the discrete formulation chosen.

II.1-c Mask System

OPA works with interior land and topography areas, although the computations occur over the entire model domain. The process of defining which areas are to be masked is described in § III.2-b, while this section describes how the masking affects the computation of the various terms of the equations, especially the boundary condition at solid walls.

The discrete representation of a domain with complex boundaries (coastlines and bottom topography) leads to arrays that include large portions where a computation is not required as the model variables remain at zero. Nevertheless, vectorial supercomputers are far more efficient when computing over a whole array, and the readability of a code is greatly improved when boundary conditions are applied in an automatic way rather than by a specific computation before or after each do loop. An efficient way to work over the whole domain while specifying the boundary conditions is to use the multiplication by mask arrays in the computation. A mask array is a matrix which elements are *1* in the ocean domain and *0* elsewhere. A simple multiplication of a variable by its own mask ensures that it will remain zero over land areas. Since most of the boundary conditions consist of a zero flux across the solid boundaries, they can be simply settled by multiplying variables by the right mask arrays, i.e. the mask array of the grid point where the flux is evaluated. For example, the heat flux in the **i**-direction is evaluated at *u*-points. Evaluating this quantity as,

$$\frac{A^{lT}}{e_1} \frac{\partial T}{\partial i} \equiv \frac{A_u^{lT}}{e_{1u}} \delta_{i+1/2}[T]\ \text{mask}_u \quad \text{(II.1.12)}$$

where mask_u is the mask array at *u*-point, ensures that the heat flux is zero inside land and at the boundaries as mask_u is zero at solid boundaries defined at *u*-points in

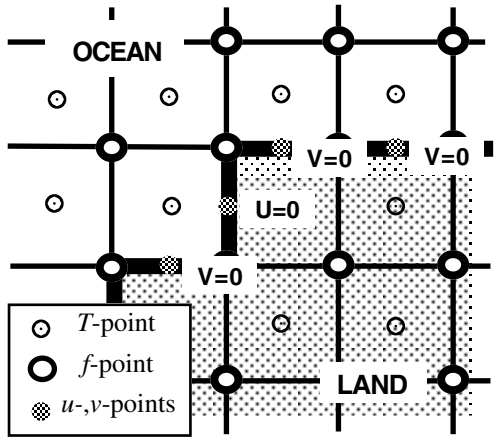


Figure II.2 Lateral boundary (thick line) at T-level. The velocity normal to the boundary is set to zero.

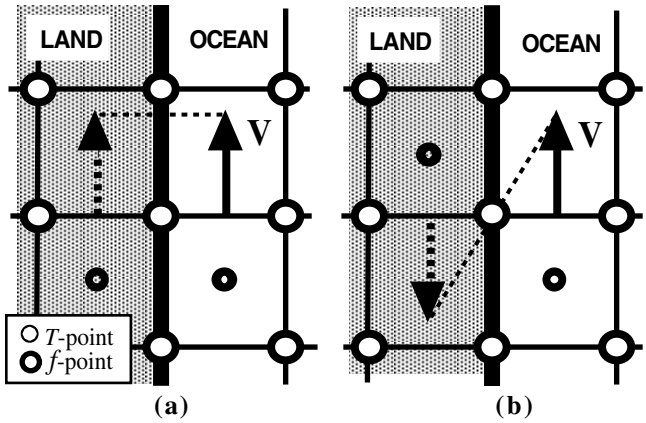


Figure II.3 (a) free-slip and (b) no-slip lateral boundary conditions.

this case (normal velocity u remains zero at the coast) (Fig.II.2).

On momentum the situation is a bit more complex as two boundary conditions must be provided along the coast (one on the normal velocity and the other on the tangential velocity). The boundary of the ocean in C-grid is defined by the velocity-faces. For example, at a given T-level , the lateral boundary (coastline or intersection with the bottom topography) is made of segments joining f-points, and normal velocity points are located between two f-points (Fig.II.2). The boundary condition on the normal velocity (no flux through solid boundaries) can thus be easily settled by the mask system. The boundary condition on the tangential velocity requires a more specific treatment. It influences the relative vorticity and momentum diffusive trends, and is only required to compute the vorticity at the coast:

- free-slip boundary condition: the normal derivative of the tangential velocity is zero at the coast, so the vorticity: mask_f array is set to zero inside the land and at the coast.

- no-slip boundary condition: the tangential velocity vanishes at the coastline. Assuming that the tangential velocity decreases linearly from the closest ocean velocity grid point to the coastline, the normal derivative is evaluated as if the closest land velocity gridpoint were of the same magnitude as the closest ocean velocity gridpoint but in the opposite direction (Fig.II.3-b). Therefore, the vorticity along the coastlines is given by: $\zeta \equiv 2\,(\,\delta_{i+1/2}\left[e_{2v}\,v\right] - \delta_{j+1/2}\left[e_{1u}\,u\right]\,)\big/(e_{1f}\,e_{2f}\,)$, where u and v are masked fields. Setting the mask_f array to *2* along the coastline allows to provide a vorticity field computed with the no-slip boundary condition simply by multiplying it by the mask_f :

$$\zeta \equiv \frac{1}{e_{1f}\,e_{2f}}\left(\delta_{i+1/2}\left[e_{2v}\,v\right] - \delta_{j+1/2}\left[e_{1u}\,u\right]\right)\text{mask}_f \qquad \text{(II.1.13)}$$

A partial- or a strong-slip boundary condition can be set along the coastline. This is equivalent to making another assumption on the velocity profile in the coastline vicinity. This can be settled by providing a value of mask_f strictly inbetween *0* and *2* (partial-slip), or larger than *2* (strong-slip).

The nature of the lateral boundary condition is controled by the value of *shlat* (**namelist** parameter) which is equal to the value of the mask_f array along the coastline, i.e. *shlat=0* for free-slip, *shlat=2* for no-slip, *0<shlat<2* for partial-slip, and *shlat>2* for strong-slip. Note that when the bottom topography is entirely represented by the s-coordinates, the lateral boundary condition on momentum tangential velocity is of little importance as it is only applied next to the coast where the minimum water depth can be quite shallow.

II.2 SEMI-DISCRETE SPACE EQUATIONS

II.2-a Ocean Dynamics

Using the representation described in § II.1, several semi-discrete space forms of the dynamical equations are available depending on the vertical coordinate used and on the conservative properties of the vorticity term. In this section, we first provide the semi-discrete space form of the dynamical equations in z- and s-coodinates with a vorticity term formulation that conserves total potential enstrophy. The two other available forms of the vorticity term are then given for both types of vertical coordinates. In all the equations, the masking has been omitted for simplicity. One must be aware that all the quantities are masked fields and that each time a mean or difference operator is used, the resulting field is multiplied by a mask.

*** *z*-coordinate dynamical equations:**

The semi-discrete space form of the momentum equations on the staggered mesh can be written in *z*-coordinate as follows:

$$\begin{aligned}\frac{\partial u}{\partial t} = &+\frac{1}{e_{1u}}\overline{(\zeta+f)}^{\,j}\,\overline{\overline{(e_{1v}v)}}^{\,i+1/2,j}-\frac{1}{2e_{1u}}\delta_{i+1/2}\left[\overline{u^2}^{\,i}+\overline{v^2}^{\,j}\right]\\ &-\frac{1}{e_{1u}e_{2u}e_{3u}}\overline{\overline{e_{1T}e_{2T}w}^{\,i+1/2}\delta_{k+1/2}[u]}^{\,k}\\ &-\frac{1}{\rho_o e_{1u}}\delta_{i+1/2}[p^h]-\left(M_u+\frac{1}{H_u e_{2u}}\delta_j\left(\frac{\partial\psi}{\partial t}\right)\right)\\ &+D_u^{vI}+D_u^{vU}\end{aligned} \tag{II.2.1}$$

$$\begin{aligned}\frac{\partial v}{\partial t} = &-\frac{1}{e_{2v}}\overline{(\zeta+f)}^{\,i}\,\overline{\overline{(e_{2u}u)}}^{\,i,j+1/2}-\frac{1}{2e_{2v}}\delta_{j+1/2}\left[\overline{u^2}^{\,i}+\overline{v^2}^{\,j}\right]\\ &-\frac{1}{e_{1v}e_{2v}e_{3v}}\overline{\overline{e_{1T}e_{2T}w}^{\,j+1/2}\delta_{k+1/2}[v]}^{\,k}\\ &-\frac{1}{\rho_o e_{2v}}\delta_{j+1/2}[p^h]-\left(M_v-\frac{1}{H_v e_{1v}}\delta_i\left(\frac{\partial\psi}{\partial t}\right)\right)\\ &+D_v^{vI}+D_v^{vU}\end{aligned} \tag{II.2.2}$$

$$\begin{aligned}&\delta_{i+1/2}\left[\frac{e_{2v}}{H_v e_{1v}}\delta_i\left[\frac{\partial\psi}{\partial t}\right]\right]+\delta_{j+1/2}\left[\frac{e_{1u}}{H_u e_{2u}}\delta_j\left[\frac{\partial\psi}{\partial t}\right]\right]\\ &\qquad=\delta_{i+1/2}\left[e_{2v}M_v\right]-\delta_{j+1/2}\left[e_{1u}M_u\right]\end{aligned} \tag{II.2.3}$$

where (M_u, M_v) represent the collected contributions of the nonlinear, viscous and hydrostatic pressure gradient terms in (II.2.1) and (II.2.2) vertically averaged over the whole water column, i.e. using the discrete vertical mean operator given by (II.1.7). (D_u^{vI}, D_v^{vI}) and (D_u^{vU}, D_v^{vU}) are the discrete formulations of the lateral and vertical momentum physics, respectively. They are given in § III.6 and § III.7. The hydrostatic pressure can be computed from (I.3.15). Nevertheless, the pressure is quite large at great depth while its horizontal gradient is several orders of magnitude smaller. This may lead to large truncation errors in the pressure gradient terms. Thus, the two horizontal components of the hydrostatic pressure gradient are computed directly as follows :

for k=km (surface layer)

$$\begin{cases}\delta_{i+1/2}[p^h]\big|_{k=km}=\dfrac{1}{2}g\,\delta_{i+1/2}[e_{3w}\rho]\big|_{k=km}\\ \delta_{j+1/2}[p^h]\big|_{k=km}=\dfrac{1}{2}g\,\delta_{j+1/2}[e_{3w}\rho]\big|_{k=km}\end{cases} \tag{II.2.4}$$

for $1<k<km$ (interior layer)

$$\begin{cases}\delta_{i+1/2}[p^h]\big|_k=\delta_{i+1/2}[p^h]\big|_{k-1}+\dfrac{1}{2}g\,\delta_{i+1/2}\left[e_{3w}\,\overline{\rho}^{\,k+1/2}\right]\Big|_k\\ \delta_{j+1/2}[p^h]\big|_k=\delta_{j+1/2}[p^h]\big|_{k-1}+\dfrac{1}{2}g\,\delta_{j+1/2}\left[e_{3w}\,\overline{\rho}^{\,k+1/2}\right]\Big|_k\end{cases}$$

The vertical velocity is computed as follows:

$$\begin{cases}w|_{km+1/2}=0\\ w|_{k-1/2}=w|_{k+1/2}-e_{3T}\,\chi|_k\end{cases} \tag{II.2.5}$$

Using (II.2.5), the bottom boundary condition ($w|_{bottom}=0$) is automatically achieved at least at the computer accuracy, due to the discrete expression of the surface pressure gradient (Appendix C).

The vorticity and the horizontal divergence are given by:

$$\zeta=\frac{1}{e_{1f}e_{2f}}\left(\delta_{i+1/2}\left[e_{2v}v\right]-\delta_{j+1/2}\left[e_{1u}u\right]\right) \tag{II.2.6}$$

$$\chi=\frac{1}{e_{1T}e_{2T}}\left(\delta_i\left[e_{2u}u\right]+\delta_j\left[e_{1v}v\right]\right) \tag{II.2.7}$$

*** *s*-coordinate dynamical equations:**

Formally, the *s*-coordinate semi-discrete space form of the dynamic equations only differs from the *z*- one in the two components of the momentum equation and in the horizontal divergence formulation. These modified three equations are written as follows:

$$\begin{aligned}\frac{\partial u}{\partial t} = &+\frac{1}{e_{1u}}\overline{\left(\frac{\zeta+f}{e_{3f}}\right)}^{\,j}\,\overline{\overline{(e_{1v}e_{3v}v)}}^{\,i+1/2,j}-\frac{1}{2e_{1u}}\delta_{i+1/2}\left[\overline{u^2}^{\,i}+\overline{v^2}^{\,j}\right]\\ &-\frac{1}{e_{1u}e_{2u}e_{3u}}\overline{\overline{e_{1T}e_{2T}w}^{\,i+1/2}\delta_{k+1/2}[u]}^{\,k}\\ &-\frac{1}{\rho_o e_{1u}}\delta_{i+1/2}[p^h]+\frac{g\,\overline{\rho}^{\,i+1/2}}{\rho_o e_{1u}}\delta_{i+1/2}[z_T]\\ &-\left(M_u+\frac{1}{H_u e_{2u}}\delta_j\left(\frac{\partial\psi}{\partial t}\right)\right)+D_u^{vI}+D_u^{vU}\end{aligned} \tag{II.2.8}$$

$$\begin{aligned}\frac{\partial v}{\partial t} = &-\frac{1}{e_{2v}}\overline{\left(\frac{\zeta+f}{e_{3f}}\right)}^{\,i}\,\overline{\overline{(e_{2u}e_{3u}u)}}^{\,i,j+1/2}-\frac{1}{2e_{2v}}\delta_{j+1/2}\left[\overline{u^2}^{\,i}+\overline{v^2}^{\,j}\right]\\ &-\frac{1}{e_{1u}e_{2u}e_{3u}}\overline{\overline{e_{1T}e_{2T}w}^{\,i+1/2}\delta_{k+1/2}[u]}^{\,k}\\ &-\frac{1}{\rho_o e_{2v}}\delta_{j+1/2}[p^h]+\frac{g\,\overline{\rho}^{\,j+1/2}}{\rho_o e_{2v}}\delta_{j+1/2}[z_T]\\ &-\left(M_v-\frac{1}{H_v e_{1v}}\delta_i\left(\frac{\partial\psi}{\partial t}\right)\right)+D_v^{vI}+D_v^{vU}\end{aligned} \tag{II.2.9}$$

$$\chi=\frac{1}{e_{1T}e_{2T}e_{3T}}\left[\delta_i\left(e_{2u}e_{3u}u\right)+\delta_j\left(e_{1v}e_{3v}v\right)\right] \tag{II.2.10}$$

Note that whereas the vertical velocity and the vorticity have the same discrete expression, their physical meanings have changed. *w* is the velocity normal to the *s*-surfaces while ζ is a pseudo vorticity along *s*-surfaces (only pseudo as (u, v) are still defined along geopotential surfaces, but not necessary defined at the same depth).

*** vorticity term in *z*- and *s*-coordinates:**
(default option or **key_vorenergy** or **key_vorcombined** defined)

The discretization of the vorticity term both in *z*- and *s*-coordinates is given in (II.2.1), (II.2.2) and (II.2.8), (II.2.9) conserves the potential enstrophy of horizontally non divergent flow [Sadourny 1975]. Other discretizations are available which conserve horizontal kinetic energy or potential enstrophy for the relative vorticity term and horizontal kinetic energy for the planetary vorticity term (see §II.4 and appendix C). All the available schemes are given just below.

The standard discrete formulation of the vorticity term provides a global conservation of the enstrophy ($(\zeta+f)^2$ in *z*-coordinates, $\left[(\zeta+f)/e_{3f}\right]^2$ in *s*-coordinates) for a horizontally non-divergent flow (i.e. $\chi=0$), but does not conserve of the total kinetic energy. It is given by:

$$z-\text{coordinate:}\quad \begin{pmatrix} -\dfrac{1}{e_{1u}}\overline{(\zeta+f)}^{\,i}\,\overline{\overline{(e_{1v}\,v)}}^{\,i,j+1/2} \\ \dfrac{1}{e_{2v}}\overline{(\zeta+f)}^{\,j}\,\overline{\overline{(e_{2u}\,u)}}^{\,i+1/2,j} \end{pmatrix}$$

$$s-\text{coordinate}\quad \begin{pmatrix} -\dfrac{1}{e_{1u}}\overline{\left(\dfrac{\zeta+f}{e_{3f}}\right)}^{\,i}\,\overline{\overline{(e_{1v}e_{3v}\,v)}}^{\,i,j+1/2} \\ \dfrac{1}{e_{2v}}\overline{\left(\dfrac{\zeta+f}{e_{3f}}\right)}^{\,j}\,\overline{\overline{(e_{2u}e_{3u}\,u)}}^{\,i+1/2,j} \end{pmatrix} \qquad \text{(II.2.11)}$$

A kinetic energy conserving scheme can be optionally used (**key_vorenergy** defined). It conserves the global kinetic energy but not the global enstrophy [Sadourny 1975]. It is given by:

$$z-\text{coordinate:}\quad \begin{pmatrix} -\dfrac{1}{e_{1u}}\overline{(\zeta+f)\overline{(e_{1v}v)}^{\,i+1/2}}^{\,j} \\ \dfrac{1}{e_{2v}}\overline{(\zeta+f)\overline{(e_{2u}u)}^{\,j+1/2}}^{\,i} \end{pmatrix}$$

$$s-\text{coordinate:}\quad \begin{pmatrix} -\dfrac{1}{e_{1u}}\overline{\left(\dfrac{\zeta+f}{e_{3f}}\right)\overline{(e_{1v}e_{3v}v)}^{\,i+1/2}}^{\,j} \\ \dfrac{1}{e_{2v}}\overline{\left(\dfrac{\zeta+f}{e_{3f}}\right)\overline{(e_{2u}e_{3u}u)}^{\,j+1/2}}^{\,i} \end{pmatrix} \qquad \text{(II.2.12)}$$

A third scheme can be optionally used (**key_vorcombined** defined). It consists of the enstrophy conserving scheme (II.2.11) applied to the relative vorticity term and of the horizontal kinetic energy conserving scheme (II.2.12) applied to the planetary vorticity term.

II.2-b Ocean Thermodynamics

*** *z*-coordinate tracer equations:**

The semi-discrete space form of the tracer equations on the staggered mesh can be written in *z*-coordinate as follows:

$$\begin{aligned} \frac{\partial T}{\partial t} &= -\frac{1}{e_{1T}e_{2T}}\left(\delta_i\left[e_{2u}\overline{T}^{\,i+1/2}u\right]+\delta_j\left[e_{1v}\overline{T}^{\,j+1/2}v\right]\right) \\ &\quad -\frac{1}{e_{3T}}\delta_k\left[\overline{T}^{\,k+1/2}w\right]+D_T^{lT}+D_T^{vT} \\ \frac{\partial S}{\partial t} &= -\frac{1}{e_{1T}e_{2T}}\left(\delta_i\left[e_{2u}\overline{S}^{\,i+1/2}u\right]+\delta_j\left[e_{1v}\overline{S}^{\,j+1/2}v\right]\right) \\ &\quad -\frac{1}{e_{3T}}\delta_k\left[\overline{S}^{\,k+1/2}w\right]+D_T^{lS}+D_T^{vS} \end{aligned} \qquad \text{(II.2.13)}$$

*** *s*-coordinate tracer equations:**

Formally, the *s*-coordinate semi-discrete space form of the tracer equations only differs from the *z*- one due to the change in the discrete form of the divergence. They can be written as follows:

$$\begin{aligned} \frac{\partial T}{\partial t} &= -\frac{1}{e_{1T}e_{2T}e_{3T}}\left(\delta_i\left[e_{2u}e_{3u}\overline{T}^{\,i+1/2}u\right]+\delta_j\left[e_{1v}e_{3v}\overline{T}^{\,j+1/2}v\right]\right) \\ &\quad -\frac{1}{e_{3T}}\delta_k\left[\overline{T}^{\,k+1/2}w\right]+D_T^{lT}+D_T^{vT} \\ \frac{\partial S}{\partial t} &= -\frac{1}{e_{1T}e_{2T}e_{3T}}\left(\delta_i\left[e_{2u}e_{3u}\overline{S}^{\,i+1/2}u\right]+\delta_j\left[e_{1v}e_{3v}\overline{S}^{\,j+1/2}v\right]\right) \\ &\quad -\frac{1}{e_{3T}}\delta_k\left[\overline{S}^{\,k+1/2}w\right]+D_T^{lS}+D_T^{vS} \end{aligned} \qquad \text{(II.2.14)}$$

(D_T^{lT} , D_T^{lS}) and (D_T^{vT} , D_T^{vS}) are the discrete formulations of the lateral and vertical tracer physics, respectively. They are given in § III.6 and § III.7.

II.2-c Ocean Physics

*** lateral subgrid scale physics on tracers:**

The general form of the second order lateral tracer subgrid scale physics (I.5.2) takes the following semi-discrete space form in *z*- and *s*-coordinates:

$$\begin{aligned} D_T^{lT} = \frac{1}{e_{1T}e_{2T}e_{3T}}\Bigg\{ &\delta_i\left[A_u^{lT}\left(\frac{e_{2u}e_{3u}}{e_{1u}}\delta_{i+1/2}[T]-e_{2u}r_{1u}\overline{\overline{\delta_{k+1/2}[T]}}^{\,i+1/2,k}\right)\right] \\ &+\delta_j\left[A_v^{lT}\left(\frac{e_{1v}e_{3v}}{e_{2v}}\delta_{j+1/2}[T]-e_{1v}r_{2v}\overline{\overline{\delta_{k+1/2}[T]}}^{\,j+1/2,k}\right)\right] \\ &+\delta_k\Bigg[A_w^{lT}\Bigg(-e_{2w}r_{1w}\overline{\overline{\delta_{i+1/2}[T]}}^{\,i,k+1/2}-e_{1w}r_{2w}\overline{\overline{\delta_{j+1/2}[T]}}^{\,j,k+1/2} \\ &+\frac{e_{1w}e_{2w}}{e_{3w}}\left(r_{1w}^{\,2}+r_{2w}^{\,2}\right)\delta_{k+1/2}[T]\Bigg)\Bigg]\Bigg\} \end{aligned} \qquad \text{(II.2.15)}$$

where r_1 and r_2 are the slopes between the surface of computation (*z*- or *s*-surfaces) and the surface along which the diffusive operator acts. The way these slopes are evaluated is given in § III.6. In the case of Gent and McWilliams [1990] parameterization, an additional tracer advection is used. Its formulation depends on r_1 and r_2 . It is also given in § III.6. At the surface, bottom and lateral boundaries, the turbulent fluxes of heat and salt are set to zero. In the special case of diffusion acting along model levels, (II.2.17) reduces to:

$$D_T^{lT} = \frac{1}{e_{1T}\, e_{2T}} \left[\delta_i \left[A_u^{eiv} \left(\frac{e_{2u}}{e_{1u}}\, \delta_{i+1/2}[T] \right) \right] + \delta_j \left[A_v^{eiv} \left(\frac{e_{1v}}{e_{2v}}\, \delta_{j+1/2}[T] \right) \right] \right] \tag{II.2.16}$$

The lateral fourth order operator formulation on tracers is simply obtained by applying (II.2.15) or (II.2.16) twice. It requires an additional assumption on boundary conditions: first and third derivative terms normal to the coast, the bottom and the surface are set to zero.

*** lateral subgrid scale physics on momentum:**

For geopotential diffusion in *z*-coordinates: the second order lateral momentum subgrid scale physics takes the following semi-discrete space form :

$$\begin{aligned} D_u^{lU} &= \frac{1}{e_{1u}}\, \delta_{i+1/2}\left[A_T^{lm}\, \chi \right] - \frac{1}{e_{2u}}\, \delta_j\left[A_f^{lm}\, \zeta \right] \\ D_v^{lU} &= \frac{1}{e_{2v}}\, \delta_{j+1/2}\left[A_T^{lm}\, \chi \right] + \frac{1}{e_{1v}}\, \delta_i\left[A_f^{lm}\, \zeta \right] \end{aligned} \tag{II.2.17}$$

For lateral diffusion along *s*-surfaces in *s*-coordinates, the operator becomes :

$$\begin{aligned} D_u^{lU} &= \frac{1}{e_{1u}}\, \delta_{i+1/2}\left[A_T^{lm}\, \chi \right] - \frac{1}{e_{2u}\, e_{3u}}\, \delta_j\left[A_f^{lm}\, e_{3f}\, \zeta \right] \\ D_v^{lU} &= \frac{1}{e_{2v}}\, \delta_{j+1/2}\left[A_T^{lm}\, \chi \right] + \frac{1}{e_{1v}\, e_{3v}}\, \delta_i\left[A_f^{lm}\, e_{3f}\, \zeta \right] \end{aligned} \tag{II.2.18}$$

When a rotation of lateral momentum diffusive operator is used (i.e. for geopotential or isopycnal diffusion in *s*-coordinates or for isopycnal diffusion in *z*-coordinates), the diffusive operator is similar to (II.2.15) applied to each horizontal component of the velocity, that is:

$$\begin{aligned} D_u^{lU} = \frac{1}{e_{1u}\, e_{2u}\, e_{3u}} \Bigg\{ & \delta_{i+1/2}\left[A_T^{lm}\, e_{2T} \left(\frac{e_{3T}}{e_{1T}}\, \delta_i[u] - r_{1T}\, \overline{\overline{\delta_{k+1/2}[u]}}^{\,i,k} \right) \right] \\ & + \delta_j\left[A_f^{lm}\, e_{1f} \left(\frac{e_{3f}}{e_{2f}}\, \delta_{j+1/2}[u] - r_{2f}\, \overline{\overline{\delta_{k+1/2}[u]}}^{\,j+1/2,k} \right) \right] \\ & + \delta_k\Bigg[A_{uw}^{lm} \Big(-e_{2u}\, r_{1uw}\, \overline{\overline{\delta_{i+1/2}[u]}}^{\,i+1/2,k+1/2} - e_{1u}\, r_{2uw}\, \overline{\overline{\delta_{j+1/2}[u]}}^{\,j,k+1/2} \\ & \qquad + \frac{e_{1u}\, e_{2u}}{e_{3uw}} \left({r_{1uw}}^2 + {r_{2uw}}^2 \right) \delta_{k+1/2}[u] \Big) \Bigg] \Bigg\} \end{aligned} \tag{II.2.19}$$

$$\begin{aligned} D_v^{lU} = \frac{1}{e_{1v}\, e_{2v}\, e_{3v}} \Bigg\{ & \delta_{i+1/2}\left[A_f^{lm}\, e_{2f} \left(\frac{e_{3f}}{e_{1f}}\, \delta_{i+1/2}[v] - r_{1f}\, \overline{\overline{\delta_{k+1/2}[v]}}^{\,i+1/2,k} \right) \right] \\ & + \delta_j\left[A_T^{lm}\, e_{1T} \left(\frac{e_{3T}}{e_{2T}}\, \delta_j[v] - r_{2T}\, \overline{\overline{\delta_{k+1/2}[v]}}^{\,j,k} \right) \right] \\ & + \delta_k\Bigg[A_{vw}^{lm} \Big(-e_{2v}\, r_{1vw}\, \overline{\overline{\delta_{i+1/2}[v]}}^{\,i,k+1/2} - e_{1v}\, r_{2vw}\, \overline{\overline{\delta_{j+1/2}[v]}}^{\,j+1/2,k+1/2} \\ & \qquad + \frac{e_{1v}\, e_{2v}}{e_{3vw}} \left({r_{1vw}}^2 + {r_{2vw}}^2 \right) \delta_{k+1/2}[v] \Big) \Bigg] \Bigg\} \end{aligned}$$

where r_1 and r_2 are the slopes between the surface along which the diffusive operator acts and the surface of computation (*z*- or *s*-surfaces). The way these slopes are evaluated is given in § III.6.

The lateral fourth order operator formulation on momentum is obtained by applying (II.2.17) or (II.2.18) or (II.2.19) twice. It requires an additional assumption on boundary conditions: first derivative term normal to the coast is depending on the free or no-slip lateral boundary conditions chosen, while the third derivative terms normal to the coast are set to zero (see § III.6).

*** vertical subgrid scale physics:**

The formulation of the vertical subgrid scale physics is the same in *z*- and *s*-coordinate as the horizontal scale factors do not depend on the vertical coordinates used (see § I.3-a). The vertical diffusive operators given by (I.5.1) take the following semi-discrete space form:

$$\begin{aligned} D_u^{vm} &\equiv \frac{1}{e_{3u}}\, \delta_k\left[\frac{A_{uw}^{vm}}{e_{3uw}}\, \delta_{k+1/2}[u] \right] \\ D_v^{vm} &\equiv \frac{1}{e_{3v}}\, \delta_k\left[\frac{A_{vw}^{vm}}{e_{3vw}}\, \delta_{k+1/2}[v] \right] \\ D_T^{vT} &\equiv \frac{1}{e_{3T}}\, \delta_k\left[\frac{A_w^{vT}}{e_{3w}}\, \delta_{k+1/2}[T] \right], \\ D_T^{vS} &\equiv \frac{1}{e_{3T}}\, \delta_k\left[\frac{A_w^{vT}}{e_{3w}}\, \delta_{k+1/2}[S] \right] \end{aligned} \tag{II.2.20}$$

where A_{uw}^{vm}, A_{vw}^{vm} and A_w^{vT} are the vertical eddy viscosity and diffusivity coefficients. The way these coefficients can be evaluated is given in § III.7.

At surface and bottom boundaries, the turbulent fluxes of momentum, heat and salt must be specified. At the surface they are prescribed from the surface forcing (see § III.4), while at the bottom they are set to zero for heat and salt, and specified through a bottom friction parameterization (see § III.9).

Note that when the ocean is coupled to a sea-ice model, the surface boundary condition on temperature switches from a Neuman to a Dirichlet boundary condition: the surface heat flux is no more specified, but diagnosed from the setting of the freezing temperature at the first ocean level.

References

(see OPA-Bibliography when the year is in **bold**)

Gent, P. R., and J. C. McWilliams, 1990: Isopycnal mixing in ocean circulation models. *J. Phys. Oceanogr., 20*, 150-155.

Sadourny, R., 1975: The dynamics of finite-difference models of the shallow-water equations. *J. Atmos. Sci., 32*, 680-689.

II.3 TIME OPERATORS

II.3-a Non-Diffusive Part—Leapfrog Scheme

The time differencing scheme used in OPA for non-diffusive processes is the well known leapfrog (three-level centered) scheme :

$$u^{t+\Delta t} = u^{t-\Delta t} + 2\Delta t \; RHS^{t} \tag{II.3.1}$$

where *RHS* is the non-diffusive right hand side of a given equation, Δt is the time step and the overscripts indicate the time at which a quantity is evaluated. The first time step of this three level scheme when starting from an ocean at rest (not from a restart file which provides the two previous time steps, see §III.12) is a forward step $u^{1} = u^{o} + \Delta t \; RHS^{o}$.

This scheme is widely used for advective processes in low-viscosity fluids. It is an efficient method that achieves second-order accuracy with just one right hand side evaluation per time step. Moreover, it does not artificially damp linear oscillatory motion nor does it produce instability by amplifying the oscillations. These advantages are somewhat diminished by the large phase-speed error of the leapfrog scheme, and the unsuitability of leapfrog differencing for the representation of diffusive and Rayleigh damping processes. However, the most serious problem associated with the leapfrog scheme is a high-frequency computational noise called "time-splitting" [Haltiner and Williams 1980] that develops when the method is used to model non linear fluid dynamics: the even and odd time steps tend to diverge between a physical and a computational mode. Time splitting can be controlled through the use of an Asselin time filter (first designed by Robert [1966] and more comprehensively studied by Asselin [1972]) or by periodically reinitialising the leapfrog solution through a single integration step with a two-level scheme. In OPA we follow the first strategy:

$$u_{f}^{t} = u^{t} + \gamma \left[u_{f}^{t-\Delta t} - 2u^{t} + u^{t+\Delta t} \right] \tag{II.3.2}$$

where the subscript *f* denotes filtered values and γ is the asselin coefficient. γ is initialized as *atfp* (**namelist** parameter). Its default value is *ratf=0.1*. Both strategies do, nevertheless, degrade the accuracy of the calculation from second to first order. The leapfrog scheme associated to a Robert-Asselin time filter has been prefered to other time differencing schemes such as predictor corrector or trapezoidal schemes because the user can control the magnitude and the spatial structure of the time diffusion of the scheme. This is exactly the same strategy as for the space discretization of the advective terms of momentum and tracer equations. The choice made is to avoid implicit numerical diffusion in both the time and space discretisation of the advective term, and add an explicit diffusive operator.

II.3-b Diffusive Part—Forward or Backward Scheme

The leapfrog differencing is unsuitable for the representation of diffusive and damping processes. For *D*, a horizontal diffusive terms and/or the restoring terms to a tracer climatology (when they are present, see § III.11), a forward time differencing scheme is used :

$$u^{t+\Delta t} = u^{t-\Delta t} + 2\Delta t \; D^{t-\Delta t} \tag{II.3.3}$$

This is diffusive in time and conditionally stable. For example, the condition of stability for a second order horizontal diffusion is:

$$A^{h} \leq e^{2} / (\pi^{2} 2\Delta t) \tag{II.3.4}$$

where *e* is the smallest grid size and A^{h} the diffusive coefficient.

For the vertical diffusion terms, a forward time differencing scheme can be used, but usually the numerical stability condition implies a strong constraint on the time step. Two solutions are available in OPA to overcome the stability constraint: (1) a forward time differencing scheme using a time splitting technique (**key_zdfexplicit** defined) or (2) a backward (or implicit) time differencing scheme by default. In (1), the master time step Δt is cut into *N* fractional time steps so that the stability criterion is reduced by a factor of *N*. The computation is done as follows :

for *L=1* to *N*

$$u_{*}^{\,t-\Delta t+L\frac{2\Delta t}{N}} = u_{*}^{\,t-\Delta t+(L-1)\frac{2\Delta t}{N}} + \frac{2\Delta t}{N} \; D^{\,t-\Delta t+(L-1)\frac{2\Delta t}{N}} \tag{II.3.5}$$

with $u_{*}^{t-\Delta t} = u^{t-\Delta t}$, $u_{*}^{t+\Delta t} = u^{t+\Delta t}$ and *D* a vertical diffusion term. The number of fractional time steps, *N*, is given by setting *navmt*, (**namelist** parameter). The scheme (2) is unconditionally stable but diffusive. It can be written as follows:

$$u^{t+\Delta t} = u^{t-\Delta t} + 2\Delta t \; D^{t+\Delta t} \tag{II.3.6}$$

This scheme is rather time consuming since it requires a matrix inversion, but it becomes attractive since a splitting factor of 3 or more is needed for the forward time differencing scheme. For example, the finite difference approximation of the temperature equation is :

$$\frac{T(k)^{t+1} - T(k)^{t-1}}{2\Delta t} \equiv RHS + \frac{1}{e_{3T}} \delta_{k} \left[\frac{A_{w}^{vT}}{e_{3w}} \delta_{k+1/2} \left[T^{t+1} \right] \right] \tag{II.3.7}$$

where RHS is the right hand side of the equation except the vertical diffusion term. Defining $c(k) = A_w^{vT}(k)/e_{3w}(k)$, $d_k = e_{3T}(k)/2\Delta t + c_k + c_{k+1}$ and $b(k) = e_{3T}(k)\,(T^{t-1}(k)/2\Delta t + RHS)$ we rewrite (II.3.7) as:

$$-c(k+1)\,T^{t+1}(k+1) + d(k)\,T^{t+1}(k) - c(k)\,T^{t+1}(k-1) \equiv b(k) \qquad \text{(II.3.8)}$$

(II.3.8) is a linear system of equations. All the elements of the corresponding matrix vanish except those on the diagonals. Moreover, $c(k)$ and $d(k)$ are positive and the diagonal term is greater than the sum of the two extra-diagonal terms, therefore a special adaptation of the Gauss elimination procedure is used to find the solution (see for example Richtmyer and Morton [1967]).

References
(see OPA-Bibliography when the year is in **bold**)

Asselin R., 1972: Frequency Filter for Time Integrations, *Mon. Wea. Rev.*, 100, 487-490.

Haltiner G. J. and R. T. Williams, 1980: *Numerical prediction and dynamic meteorology*. John Wiley & Sons Eds., second edition, 477pp.

Richtmyer, R. D., and K.W. Morton, 1967: Difference methods for initial-value problems. Interscience Publisher, Second Edition, 405pp.

Robert, A. J., 1966:. *J. Meteor. Soc. Japan, 44*, 237.

II.4 INVARIANT OF THE EQUATIONS

The continuous equations of motion have many analytic properties. Many quantities (total mass, energy, enstrophy, etc.) are strictly conserved in the inviscid and unforced limit, while ocean physics conserve the total quantities on which they act (momentum, temperature, salinity) but dissipate their total variance (energy, enstrophy, etc.). Unfortunately, finite difference form of these equations are not guaranteed to retain all these important properties. In constructing the finite differencing schemes, we wish to ensure that certain integral constraints will be maintained. In particular, it is desirable to construct the finite difference equations so that horizontal kinetic energy and/or potential enstrophy of horizontally non divergent flow, and variance of temperature and salinity will be conserved in the absence of dissipative effects and forcing. The advantage of this approach was first pointed out by Arakawa [1966]. Arakawa showed that if integral constraints on energy are maintained, the computation will be free of the troublesome "non linear" instability originally pointed out by Phillips [1959]. A consistent formulation of the energetic properties is also extremely important in carrying out long-term numerical simulations for an oceanographic model. Such a formulation avoids systematic errors which accumulates with time [Bryan, 1969].

The general philosophy of OPA which has led to the discrete formulation presented in §II.2 and II.3 is to choose second order non-diffusive scheme for advective terms for both dynamical and tracer equations. At this level of complexity, the resulting schemes are dispersive schemes. Therefore, they require the addition of a diffusive operator to be stable. The alternative is to use diffusive schemes such as upstream or flux corrected schemes. This last option was rejected because we prefer a complete handling of the model diffusion, i.e. of the model physics rather than letting the advective scheme produces its own implicit diffusion without controlling the space and time structure of this implicit diffusion. Note that in some very specific cases as passive tracer studies, the positivity of the advective scheme is required. In that case, and in that case only, the advective scheme used for passive tracer is a flux correction scheme [Marti **1992**, Lévy **1996**, Lévy *et al* **1998**].

II.4-a Conservation Properties on Ocean Dynamics

The non linear term of the momentum equations has been split into a vorticity term, a gradient of horizontal kinetic energy and a vertical advection term. Three schemes are available for the former (see § II.2) according to the cpp variable defined (default option or **key_vorenergy** or **key_vorcombined** defined). They differ in their conservative properties (energy or enstrophy conserving scheme). The two latter terms preserve the total kinetic energy: the large scale kinetic energy is also preserved in practice. The remaining non-diffusive terms of the momentum equation (namely the hydrostatic and surface pressure gradient terms) also preserve the total kinetic energy and have no effect on the vorticity of the flow.

*** relative, planetary and total vorticity term:**
Let us define ς as either the relative, planetary and total potential vorticity, i.e. ζ/e_3, f/e_3, and $(\zeta+f)/e_3$, respectively. The continuous formulation of the vorticity term satisfies following integral constraints:

$$\int_D \mathbf{k}\cdot\frac{1}{e_3}\nabla\times\left(\varsigma\,\mathbf{k}\times\mathbf{U}_h\right)dv = 0 \qquad \text{(II.4.1a)}$$

$$\text{if } \chi = 0$$
$$\int_D \varsigma\,\mathbf{k}\cdot\frac{1}{e_3}\nabla\times\left(\varsigma\mathbf{k}\times\mathbf{U}_h\right)dv = -\int_D \frac{1}{2}\varsigma^2\,\chi\,dv = 0 \qquad \text{(II.4.1b)}$$

$$\int_D \mathbf{U}_h\times\left(\varsigma\,\mathbf{k}\times\mathbf{U}_h\right)dv = 0 \qquad \text{(II.4.1c)}$$

where $dv = e_1\,e_2\,e_3\,di\,dj\,dk$ is the volume element. (II.4.1a) means that ς is conserved. (II.4.1b) is obtained by an integration by part. It means that ς^2 is conserved for a horizontally non-divergent flow. (II.4.1c) is even satisfied locally since the vorticity term is orthogonal to the horizontal velocity. It means that the vorticity term has no contribution to the evolution of the total kinetic energy. (II.4.1a) is obviously always satisfied, but (II.4.1b) and (II.4.1c) cannot be satisfied simultaneously with a second order scheme [Sadourny 1975]. Using the symmetry or anti-symmetry properties of the operators (Eqs II.1.10 and 11), it can be shown that the scheme (II.2.11) satisfies (II.4.1b) but not (II.4.1c), while scheme (II.2.12) satisfies (II.4.1c) but not (II.4.1b) (see appendix C). Note that the enstrophy conserving scheme on total vorticity has been chosen as the standard discrete form of the vorticity term.

*** Gradient of kinetic energy / vertical advection**

In continuous formulation, the gradient of horizontal kinetic energy has no contribution to the evolution of the vorticity as the curl of a gradient is zero. This property is satisfied locally with the discrete form of both the gradient and the curl operator we have made (property (II.1.9)). Another continuous property is that the change of horizontal kinetic energy due to vertical advection is exactly balanced by the change of horizontal kinetic energy due to the horizontal gradient of horizontal kinetic energy:

$$\int_D \mathbf{U}_h\cdot\nabla_h\left(1/2\,\mathbf{U}_h{}^2\right)dv = -\int_D \mathbf{U}_h\cdot\frac{w}{e_3}\frac{\partial\mathbf{U}_h}{\partial k}\,dv \qquad \text{(II.4.2)}$$

Using the discrete form given in §II.2-a and the symmetry or anti-symmetry properties of the mean and difference operators, (II.4.2) is demonstrated in the Appendix C. The main point here is that satisfying (II.4.2) links the choice of the discrete forms of the vertical advection and of the horizontal gradient of horizontal kinetic energy. Choosing one imposes the other. The discrete form of the vertical advection given in §II.2-a is a direct consequence of formulating the horizontal kinetic energy as $1/2(\overline{u^2}^i + \overline{v^2}^j)$ in the gradient term.

*** hydrostatic pressure gradient term**

In continuous formulation, a pressure gradient has no contribution to the evolution of the vorticity as the curl of a gradient is zero. This properties is satisfied locally with the choice of discretization we have made (property (II.1.9)). In addition, when the equation of state is linear (i.e. when an advective-diffusive equation for density can be derived from those of temperature and salinity) the change of horizontal kinetic energy due to the work of pressure forces is balanced by the change of potential energy due to buoyancy forces:

$$\int_D -\frac{1}{\rho_o}\nabla p^h\big|_z\cdot\mathbf{U}_h\,dv = \int_D \nabla.\left(\rho\,\mathbf{U}\right)g\,z\;dv \qquad \text{(II.4.3)}$$

Using the discrete form given in § II.2-a and the symmetry or anti-symmetry properties of the mean and difference operators, (II.4.3) is demonstrated in the Appendix C. The main point here is that satisfying (II.4.3) strongly constraints the discrete expression of the depth of *T*-points and of the term added to the pressure gradient in *s*-coordinates: the depth of a *T*-point, z_T, is defined as the sum the vertical scale factors at *w*-points starting from the surface.

*** surface pressure gradient term**

In continuous formulation, the surface pressure gradient has no contribution to the evolution of vorticity. This properties is trivially satisfied locally as (II.2.3) (the equation verified by ψ) has been derived from the discrete formulation of the momentum equations, vertical sum and curl. Nevertheless, the ψ-equation is solved numerically by an iterative solver (see § III.5), thus the property is only satisfied with the accuracy required on the solver. In addition, with the rigid-lid approximation, the change of horizontal kinetic energy due to the work of surface pressure forces is exactly zero:

$$\int_D -\frac{1}{\rho_o}\nabla_h\left(p_s\right)\cdot\mathbf{U}_h\,dv = 0 \qquad \text{(II.4.4)}$$

(II.4.4) is satisfied in discrete form only if the discrete barotropic streamfunction time evolution equation is given by (II.2.3) (see appendix C). This shows that (II.2.3) is the only way to compute the streamfunction, otherwise there is no guarantee that the surface pressure force work vanishes.

II.4-b Conservation Properties on Ocean Thermodynamics

In continuous formulation, the advective terms of the tracer equations conserve the tracer content and the quadratic form of the tracer, i.e.

$$\int_D \nabla.\left(T\,\mathbf{U}\right)dv = 0 \quad \text{and} \quad \int_D T\,\nabla.\left(T\,\mathbf{U}\right)dv = 0 \qquad \text{(II.4.5)}$$

The numerical scheme used (§II.2-b) (equations in flux form, second order centred finite differences) satisfies (II.4.5) (see appendix C). Note that in both continuous and discrete formulations, there is generally no strict conservation of mass, since the equation of state is non linear with respect to *T* and *S*. In practice, the mass is conserved with a very good accuracy.

II.4-c Conservation Properties on Momentum Physics

*** lateral momentum diffusion term**

The continuous formulation of the horizontal diffusion of momentum satisfies the following integral constraints :

$$\int_D \frac{1}{e_3}\mathbf{k}\cdot\nabla\times\left[\nabla_h\left(A^{lm}\,\chi\right)-\nabla_h\times\left(A^{lm}\,\zeta\,\mathbf{k}\right)\right]dv=0 \qquad \text{(II.4.6a)}$$

$$\int_D \nabla_h\cdot\left[\nabla_h\left(A^{lm}\,\chi\right)-\nabla_h\times\left(A^{lm}\,\zeta\,\mathbf{k}\right)\right]dv=0 \qquad \text{(II.4.6b)}$$

$$\int_D \mathbf{U}_h\cdot\left[\nabla_h\left(A^{lm}\,\chi\right)-\nabla_h\times\left(A^{lm}\,\zeta\,\mathbf{k}\right)\right]dv\le 0 \qquad \text{(II.4.6c)}$$

$$\text{if}\quad A^{lm}=cste$$
$$\int_D \zeta\,\mathbf{k}\cdot\nabla\times\left[\nabla_h\left(A^{lm}\,\chi\right)-\nabla_h\times\left(A^{lm}\,\zeta\,\mathbf{k}\right)\right]dv\le 0 \qquad \text{(II.4.6d)}$$

$$\text{if}\quad A^{lm}=cste$$
$$\int_D \chi\,\nabla_h\cdot\left[\nabla_h\left(A^{lm}\,\chi\right)-\nabla_h\times\left(A^{lm}\,\zeta\,\mathbf{k}\right)\right]dv\le 0 \qquad \text{(II.4.6e)}$$

(II.4.6a) and (II.4.6b) means that the horizontal diffusion of momentum conserve both the potential vorticity and the divergence of the flow, while Eqs (II.4.6c) to (II.4.6e) mean that it dissipates the energy, the enstrophy and the square of the divergence. The two latter properties are only satisfied when the eddy coefficients are horizontally uniform.

Using (II.1.8) and (II.1.9), and the symmetry or anti-symmetry properties of the mean and difference operators, it is shown that the discrete form of the lateral momentum diffusion given by (II.2.17) and (II.2.18) (i.e. diffusion along model levels) satisfies all the integral constraints (II.4.6) (see appendix C). In particular, when the eddy coefficients are horizontally uniform, a complete separation of vorticity and horizontal divergence fields is ensured, so that diffusion (dissipation) of vorticity (enstrophy) does not generate horizontal divergence (variance of the horizontal divergence) and *vice versa*. When the vertical curl of the horizontal diffusion of momentum (discrete sense) is taken, the term associated to the horizontal gradient of the divergence is zero locally. When the horizontal divergence of the horizontal diffusion of momentum (discrete sense) is taken, the term associated to the vertical curl of the vorticity is zero locally. The resulting term conserves χ and dissipates χ^2 when the eddy coefficient is horizontally uniform.

When (II.2.19) is used (i.e. when a rotation of lateral momentum diffusive operator is used) none of the above properties can be established in discrete form.

*** vertical momentum diffusion term**

As for the lateral momentum physics, the continuous form of the vertical diffusion of momentum satisfies following integral constraints :

conservation of momentum, dissipation of horizontal kinetic energy

$$\begin{aligned}&\int_D \frac{1}{e_3}\frac{\partial}{\partial k}\left(\frac{A^{vm}}{e_3}\frac{\partial \mathbf{U}_h}{\partial k}\right)dv=\mathbf{0}\\ &\int_D \mathbf{U}_h\cdot\frac{1}{e_3}\frac{\partial}{\partial k}\left(\frac{A^{vm}}{e_3}\frac{\partial \mathbf{U}_h}{\partial k}\right)dv=0\end{aligned} \qquad \text{(II.4.7a)}$$

conservation of vorticity, dissipation of enstrophy

$$\begin{aligned}&\int_D \frac{1}{e_3}\mathbf{k}\cdot\nabla\times\left(\frac{1}{e_3}\frac{\partial}{\partial k}\left(\frac{A^{vm}}{e_3}\frac{\partial \mathbf{U}_h}{\partial k}\right)\right)dv=0\\ &\int_D \zeta\,\mathbf{k}\cdot\nabla\times\left(\frac{1}{e_3}\frac{\partial}{\partial k}\left(\frac{A^{vm}}{e_3}\frac{\partial \mathbf{U}_h}{\partial k}\right)\right)dv=0\end{aligned} \qquad \text{(II.4.7b)}$$

conservation of horizontal divergence, dissipation of square of the horizontal divergence

$$\begin{aligned}&\int_D \nabla\cdot\left(\frac{1}{e_3}\frac{\partial}{\partial k}\left(\frac{A^{vm}}{e_3}\frac{\partial \mathbf{U}_h}{\partial k}\right)\right)dv=0\\ &\int_D \chi\,\nabla\cdot\left(\frac{1}{e_3}\frac{\partial}{\partial k}\left(\frac{A^{vm}}{e_3}\frac{\partial \mathbf{U}_h}{\partial k}\right)\right)dv=0\end{aligned} \qquad \text{(II.4.7c)}$$

In discrete form, all these properties are satisfied in *z*-coordinate (see Appendix C). In *s*-coordinates, only first order properties can be demonstrated, i.e. the vertical momentum physics conserve momentum, potential vorticity, and horizontal divergence.

II.4-d Conservation Properties on Tracer Physics

The numerical schemes used for tracer subgridscale physics are written in such a way that the heat and salt contents are conserved (equations in flux form, second order centred finite differences). As a form flux is used to compute the temperature and salinity, the quadratic form of these quantities (i.e. their variance) globally tends to diminish. As for the advective term, there is generally no strict conservation of mass even if, in practice, the mass is conserved with a very good accuracy.

*** lateral physics:** conservation of tracer, dissipation of tracer variance, i.e.

$$\begin{aligned}&\int_D \nabla.\left(A^{lT}\,\Re\,\nabla T\right)dv=0\\ &\int_D T\,\nabla.\left(A^{lT}\,\Re\,\nabla T\right)dv\le 0\end{aligned} \qquad \text{(II.4.8)}$$

*** vertical physics:** conservation of tracer, dissipation of tracer variance, i.e.

$$\int_D \frac{1}{e_3} \frac{\partial}{\partial k}\left(\frac{A^{vT}}{e_3} \frac{\partial T}{\partial k} \right) dv = 0$$

$$\int_D T \frac{1}{e_3} \frac{\partial}{\partial k}\left(\frac{A^{vT}}{e_3} \frac{\partial T}{\partial k} \right) dv = 0 \qquad \text{(II.4.9)}$$

Using the symmetry or anti-symmetry properties of the mean and difference operators, it is shown that the discrete form of tracer physics given in § II.2-c satisfies all the integral constraints (II.4.8) and (II.4.9) except the dissipation of the square of the tracer when non-geopotential diffusion is used (see appendix C). A discrete form of the lateral tracer physics can be derived which satisfies the last property. Nevertheless, it requires a horizontal averaging of the vertical component of the lateral physics which prevents the use of implicit time stepping scheme for the vertical diffusion term. It has not been implemented.

References

(see OPA-Bibliography when the year is in **bold**)

Arakawa A., 1966: Computational design for long term numerical integration of the equations of fluid motion, two dimensional incompressible flow, Part. I. *J. Comput. Phys., I*, 119-149.

Bryan, K., 1969: A numerical method for the study of the circulation of the world ocean. *J. Comput. Phys., 4*, 347-379.

Phillips, N. A., 1959: An example of non-linear computational instability. *The Atmosphere and the Sea motion*, Bert Bolin, ed., Rockefeller Institute Press, New York, p. 501.

Sadourny, R., 1975: The dynamics of finite-difference models of the shallow-water equations. *J. Atmos. Sci., 32*, 680-689.

III. DETAILS OF THE MODEL

III.1 NUMERICAL INDEXATION

The array representation used in the FORTRAN code requires an integer indexation while the analytical definition of the mesh (see § II.1) is associated with the use of integer values of (i, j, k) for T-points whereas all the other points use both integer and integer and a half values of (i, j, k). Therefore a specific integer indexation must be defined for the latter grid-points (i.e. velocity and vorticity grid-points).

III.1-a Horizontal Indexation

In the horizontal plane, the indexation shown in fig. III.1 has been chosen. For an increasing i index (j index), the T-point and the eastward u-point (northward v-point) have the same index (see the dashed area in fig.III.1), and a T-point and its nearby north-east f-point have the same i-and j-indices.

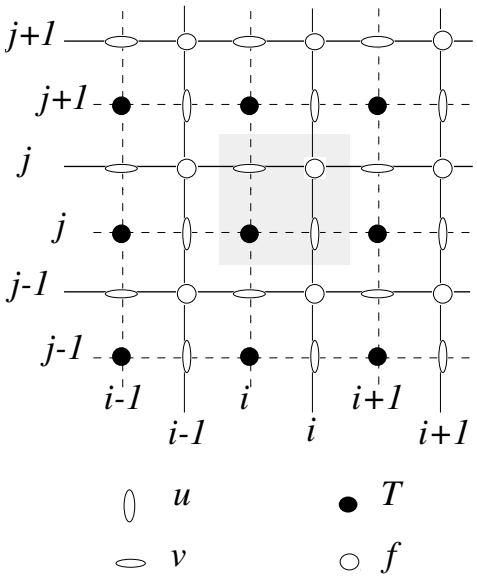


Figure III.1: horizontal integer indexation used in the FORTRAN code. The dashed area indicates the cell in which variables contained in arrays have the same i- and/or j-indices

III.1-b Vertical Indexation

In the vertical plane, the chosen indexation requires special attention since the k-axis is re-oriented downward in the FORTRAN code compared to the presentation of the discrete equations in Chapter II. The sea surface corresponds to the w-level $k=1$ like the T-level just below (Fig. III.2). The last w-level ($k=jpk$) is either the ocean bottom or inside the ocean floor while the last T-level is always inside the floor (Fig. III.2). Note that for an increasing k index, a w-point and the T-point just below have the same k index, in opposition to what is done in the horizontal plane where it is the T-point and the nearby velocity points in the direction of the horizontal axis that have the same i or j index (compare the dashed area in Fig.III.1 and III.2). As the scale factors are chosen to be strictly positive, a minus sign appears in the FORTRAN code before all the vertical derivatives of the discrete equations given in § II.3 and § II.4.

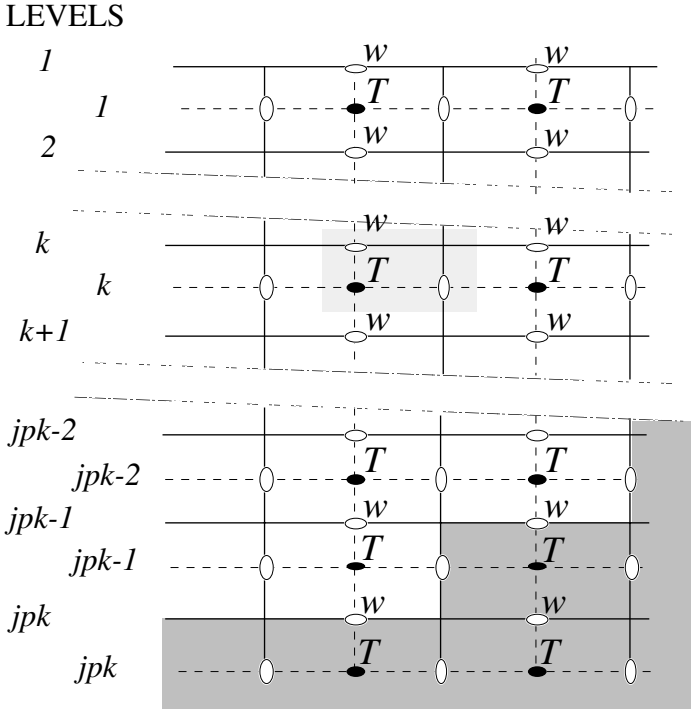


Figure III.2: vertical integer indexation used in the FORTRAN code. Note that the k-axis is oriented downward. The dashed area indicates the cell in which variables contained in arrays have the same k-index.

III.2 MODEL DOMAIN

The model domain is defined once during the initialisation phase. It consists in defining the horizontal and vertical meshes, the time stepping (from parameters given in the **namelist** file, see § III.11), the mask arrays and the islands (**key_islands** defined). All the arrays related to a particular ocean model configuration (grid-point position, scale factors, masks) can be save in a file if *nmsh=1* (**namelist** parameter). This can be particularly useful for plots and off-line diagnostics.

III.2-a Model Mesh

The ocean mesh (i.e. the position of all the scalar and vector points) is defined by the transformation that gives (λ, φ, z) as a function of (i, j, k). The grid-points are located at integer or integer and a half values of (i, j, k) of (λ, φ, z) as indicated in Table II.1 (see § II.1). The associated scale factors are defined using the analytical first derivative of the transformation in (I.3.1). These definitions are done in two routines, *domhgr.F* and *domzgr.F*, which provide the horizontal and vertical meshes, respectively. This section briefly describes how to create the two meshes.

*** horizontal mesh**

In a horizontal plane, the location of all the model grid points is defined from the analytical expressions of the latitude φ and the longitude λ as a function of (i, j). The horizontal scale factors are calculated using (I.3.1). The user must provide the analytical expression of the functions (λ, φ) and their first derivatives (λ', φ') with respect to *i* and *j*. This is done in routine *domhgr.F* as statement functions. In most applications, the latitude and longitude are function of a single value (*j* and *i*, respectively) (geographical configuration of the mesh). In this case the horizontal mesh definition reduces to define $\varphi_{(j)}$, $\varphi'_{(j)}$, $\lambda_{(i)}$ and $\lambda'_{(i)}$ as statement functions. The model computes the grid-point positions and scale factors in the horizontal plane as follows:

$$\begin{aligned} \lambda_t &\equiv glamt = \lambda_{(i)} & \varphi_t &\equiv gphit = \varphi_{(j)} \\ \lambda_u &\equiv glamu = \lambda_{(i+1/2)} & \varphi_u &\equiv gphiu = \varphi_{(j)} \\ \lambda_v &\equiv glamv = \lambda_{(i)} & \varphi_v &\equiv gphiv = \varphi_{(j+1/2)} \\ \lambda_f &\equiv glamf = \lambda_{(i+1/2)} & \varphi_f &\equiv gphif = \varphi_{(j+1/2)} \end{aligned}$$

$$\begin{aligned} e_{1t} &\equiv e1t = ra\left|\lambda'_{(i)}\cos\varphi_{(j)}\right| & e_{2t} &\equiv e2t = ra\left|\varphi'_{(j)}\right| \\ e_{1u} &\equiv e1u = ra\left|\lambda'_{(i+1/2)}\cos\varphi_{(j)}\right| & e_{2u} &\equiv e2u = ra\left|\varphi'_{(j)}\right| \\ e_{1v} &\equiv e1v = ra\left|\lambda'_{(i)}\cos\varphi_{(j+1/2)}\right| & e_{2v} &\equiv e2v = ra\left|\varphi'_{(j+1/2)}\right| \\ e_{1f} &\equiv e1f = ra\left|\lambda'_{(i+1/2)}\cos\varphi_{(j+1/2)}\right| & e_{2f} &\equiv e2f = ra\left|\varphi'_{(j+1/2)}\right| \end{aligned}$$

where the last letter of each computational name indicates the grid point considered and *ra* is the earth radius (defined in *parctl.F* along with all universal constants, see § IV.3-b). Note that the horizontal position and scale factors of *w*-points are exactly equal to those of *T*-points, thus no specific arrays are defined at those grid-points. In particular applications, such as the global ocean model [Madec and Imbard **1996**], the horizontal mesh is computed from a semi-analytical method. In that case it is read in routine *hgrcoo.F* (called in *domhgr.F* if *ngrid=1* (**namelist** parameter).

*** vertical mesh**

The vertical location of *w*- and *T*-levels is defined from an analytic expression of the depth *z* whose analytic derivative with respect to *k* provides the vertical scale factors. The formulation of the analytic expression of *z* strongly depends on the choice of the vertical coordinate. In *z*-coordinates, *z* is a function of the single value *k*, while it depends on (i, j, k) in *s*-coordinates. As for the horizontal, the user must provide the analytical expression of both *z* and its first derivative with respect to *k*. This is done in routine *domzgr.F* using statement functions, defined in include files *domzgr.z.h* or *domzgr.s.h* depending on the cpp option chosen for the vertical coordinate system (default option or **key_s_coord** defined, respectively). The following function is proposed as a standard for *z*-coordinates:

$$\begin{aligned} z_{(k)} &= h_0 - h_1\, k - h_2\, h_3 \log\left[\cosh\left((k-h_4)/h_3\right)\right] \\ e_{3(k)} &= -h_1 - h_2 \tanh\left((k-h_4)/h_3\right) \end{aligned} \qquad \text{(III.2.1)}$$

where $k = 1$ to jpk for *w*-levels and $k = 1 + 1/2$ to $k = jpk + 1/2$ for *T*-levels. Such an expression allows us to define a nearly uniform vertical location of levels at the ocean top and bottom with a smooth hyperbolic tangent transition in between (Table III.1, Fig. III.3). We have chosen a *10* m (*500* m) resolution in the surface (bottom) layers and a depth which varies from *0* at the sea surface to a minimum of *-5000* m. This leads to the following conditions:

$$\begin{aligned} e_{3(1+1/2)} &= 10. \\ e_{3(jpk-1/2)} &= 500. \\ z_{(1)} &= 0. \\ z_{(jpk)} &= -5000. \end{aligned} \qquad \text{(III.2.2)}$$

The five coefficients h_0 to h_4 in (III.2.1) are determined such that (III.2.2) is satisfied. To do so, we first make an arbitrary choice of *jpk* and h_3, here $jpk = 31$ and $h_3 = 3$. Using the first three conditions in (III.2.2), we express h_0, h_1 and h_2 as a function of h_4, and then we determine

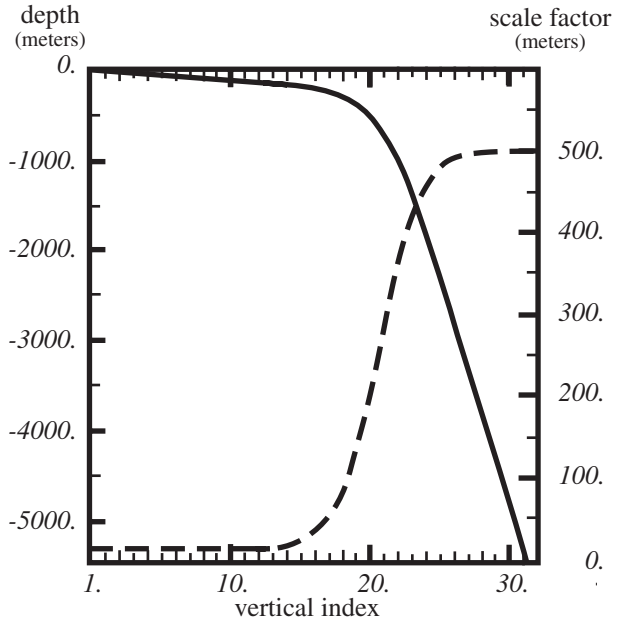


Figure III.3 : Default vertical mesh. Vertical level functions for (a) T-point depth and (b) the associated scale factor as computed from (III.2.1) in z-coordinates.

$h_4 \in [1, jpk]$ using a bisection method such that the last condition in (III.2.2) is satisfied. In the present case this leads to the following values: $h_0 = 4762.96$, $h_1 = 255.58$, $h_2 = 245.58$, and $h_4 = 21.43$. The resulting depths and scale factors as a function of the model levels are shown in Fig. III.3. An off-line program is available (OPAZGR) for defining vertical locations with numerical constraints other than (III.2.2) or a different number of vertical levels (see § IV.3).

In s-coordinates (**key_s_coord** defined), the depths of the model levels are defined from the product of a depth field and a stretching function and its derivative, respectively :

$$z(k) = h(i,j)\, z_o(k) \quad , \quad e_3(k) = h(i,j)\, z'_o(k) \qquad \text{(III.2.3)}$$

where h is the depth of the last w-level ($k = jpk$) defined at T-point location in the horizontal and z_o is a function which varies from *0* at the sea surface to *1* at the ocean bottom. The depth field h is not necessary the ocean depth as a mixed step-like and bottom following representation of the topography can be used (see § III.2-b). It is read in from file *numbat* in *dommba.F* routine. z_o and its derivative are set in *domzgr.s.h*. They are chosen by default as the function (III.2.1) divided by h_o.

Note that z and e_3 can be either a single or multiple value function. In order to minimise the change in the computer code when switching from z- to s-coordinates, they are introduced as statement functions in the include file *stafun.h*.

LEVEL	GDEPT	GDEPW	E3T	E3W
1	**5.00**	0.00	10.00	10.00
2	**15.00**	10.00	10.00	10.00
3	**25.00**	20.00	10.00	10.00
4	**35.01**	30.00	10.01	10.00
5	**45.01**	40.01	10.01	10.01
6	**55.03**	50.02	10.02	10.02
7	**65.06**	60.04	10.04	10.03
8	**75.13**	70.09	10.09	10.06
9	**85.25**	80.18	10.17	10.12
10	**95.49**	90.35	10.33	10.24
11	**105.97**	100.69	10.65	10.47
12	**116.90**	111.36	11.27	10.91
13	**128.70**	122.65	12.47	11.77
14	**142.20**	135.16	14.78	13.43
15	**158.96**	150.03	19.23	16.65
16	**181.96**	169.42	27.66	22.78
17	**216.65**	197.37	43.26	34.30
18	**272.48**	241.13	70.88	55.21
19	**364.30**	312.74	116.11	90.99
20	**511.53**	429.72	181.55	146.43
21	**732.20**	611.89	261.03	220.35
22	**1033.22**	872.87	339.39	301.42
23	**1405.70**	1211.59	402.26	373.31
24	**1830.89**	1612.98	444.87	426.00
25	**2289.77**	2057.13	470.55	459.47
26	**2768.24**	2527.22	484.95	478.83
27	**3257.48**	3011.90	492.70	489.44
28	**3752.44**	3504.46	496.78	495.07
29	**4250.40**	4001.16	498.90	498.02
30	**4749.91**	4500.02	500.00	499.54
31	**5250.23**	5000.00	500.56	500.33

Table III.1: default vertical mesh in z-coordinates as computed from (III.2.1).

III.2-b Bathymetry and Mask

Whatever the vertical coordinate used, the model offers the possibility of representing the bottom topography with steps that follow the face of the model cells (step like topography) [Madec *et al.* **1996**]. As two types of vertical coordinates can be used, the bottom topography can be represented in three discrete ways: a step-like representation in z-coordinates, or in s-coordinates, a pure terrain following representation or a hybrid representation which mixes the former two (Fig. III.4). The distribution of the steps in the horizontal is defined in a 2D integer array, *mbathy*, which gives the number of ocean levels (i.e. those that are not masked) at each T-point. If *ntopo=1* (**namelist** parameter), the 2D array is read in from a formatted file, *bathymetry*, that is constructed from the ETOPO 5'x5' global bathymetric field using the off-line program OPABAT (see § IV.3). This field is of

paramount importance. It contains all the information required to define the geometry of the coastlines, to identify the islands (**key_islands** defined), and to compute all the 3D mask fields (i.e. the step-like representation of the bottom topography).

The *mbathy* array takes values from *-N* to *jpk-1* and provides the following information: $mbathy(i,j) = -n$, $n \in]0,N]$, *T*-points are land points of the n^{th} island; $mbathy(i,j) = 0$, *T*-points are land points of the main land (continent); $mbathy(i,j) = k \in]0, jpk[$, the first *k* *T*- and *w*-points are ocean points, the others land points.

From the *mbathy* array, the mask fields are defined as follows:

$tmask(i,j,k) = 1$ or 0 whether $k \leq mbathy(i,j)$ or not

$$umask(i,j,k) = tmask(i,j,k) \times tmask(i+1,j,k)$$
$$vmask(i,j,k) = tmask(i,j,k) \times tmask(i,j+1,k)$$
$$fmask(i,j,k) = tmask(i,j,k) \times tmask(i+1,j,k) \times tmask(i,j+1,k) \times tmask(i+1,j+1,k)$$

Note that *wmask* is not defined as it is exactly equal to *tmask* with the numerical indexation used (§ III.1). Moreover, the specification of closed lateral boundaries requires that at least the first and last rows and columns of *mbathy* array are set to zero. In the particular case of an east-west cyclic boundary condition, *mbathy* has its last column equal to the second one and its first column equal to the last but one (and so the mask arrays) (see § III.10).

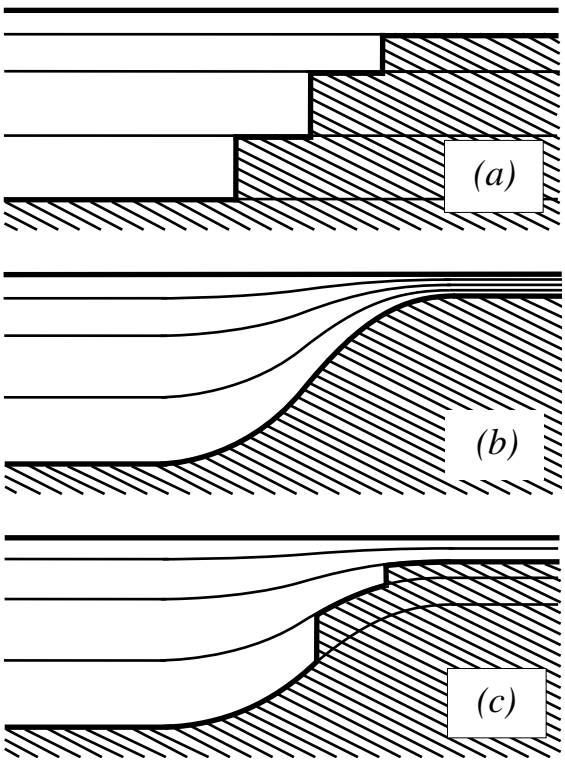


Figure III.4: The ocean bottom as seen by the model: (a) step-like representation (z-coordinates), (b) terrain following representation (s-coordinates) and (c) hybrid representation (s-coordinates).

III.3 PROPERTIES OF SEAWATER

III.3-a Equation of State

(*neos = 0, 1* or *2*, **namelist** parameter)

It is necessary to know the equation of state for the ocean very accurately to determine stability properties (especially the Brunt-Vaisälä frequency), particularly in the deep ocean. The ocean density is a non linear empirical function of *in situ* temperature, salinity and pressure. The reference is the equation of state defined by the Joint Panel on Oceanographic Tables and Standards [UNESCO, 1983]. It was the standard equation of state used in the previous release of OPA. Even though this computation is fully vectorized, it is quite time consuming (*15* to *20%* of the total CPU time) as it requires the prior computation of the *in situ* temperature from the model *potential* temperature using the Bryden [1973] polynomial for adiabatic lapse rate and a 4*th* order Runge-Kutta integration scheme. In the present release, we have chosen the Jackett and McDougall [1995] equation of state for seawater. This equation has been designed to allow the computation of the *in situ* ocean density directly as a function of *potential* temperature relative to the sea surface (an OPA variable), the practical salinity (another OPA variable) and the pressure (assuming no pressure variation along geopotential surfaces, i.e. the pressure in decibars is approximated by the depth in meters). Both the UNESCO [1983] and Jackett and McDougall [1995] equations of state have the same algebraic expression except that the values of the different coefficients have been adjusted by Jackett and McDougall [1995] in order to use directly the *potential* temperature instead of the *in situ* one. The use of this equation of state reduces the CPU time of the *in situ* density computation to about *3%* of the total CPU time, while maintaining a quite accurate evaluation of the density.

In the computer code, a *true* density, *d*, is computed, i.e. the ratio of seawater volumic mass over ρ_o, a reference volumic mass (*rau0*, **namelist** parameter). The default option (*neos=0*) is the Jackett and McDougall [1995] equation of state. It is highly recommended to use it. Nevertheless, for process studies, it is often convenient to use a linear approximation of the density. In this case, there is no longer a distinction between *in situ* and *potential* density as density does not depends on pressure

or depth. Two linear formulations are available: a function of T only (*neos=1*) and a function of both T and S (*neos=2*):

$$d(T) = \rho(T)/\rho_0 = 1.028 - \alpha\, T$$
$$d(T,S) = \rho(T,S)/\rho_0 = \beta\, S - \alpha\, T \qquad \text{(III.3.1)}$$

where α and β are the thermal and haline expansion coefficients, and ρ_o, a reference volumic mass. Their default values ($\alpha = 2.\,10^{-4}$, $\beta = 0.029$ and $\rho_o = 1020\ kg/m^3$) can be modified through *ralpha*, *rbeta* and *rau0* (**namelist** parameters). Note that when d is a function of T only (*neos=1*), the salinity is a passive tracer.

III.3-b Brunt-Vaisälä Frequency

(*neos = 0, 1* or *2*, **namelist** parameter)

An accurate computation of the ocean stability (i.e. of N, the brunt-Vaisälä frequency) is of paramount importance as it is used in several ocean parameterisations (namely the 1.5 vertical turbulent closure, Richardson number dependent vertical diffusion, enhanced vertical diffusion, non-penetrative convective adjustment, isopycnal diffusion). In particular, one must be aware that N has to be computed with an *in situ* reference, i.e. at the local pressure. The expression of N depends on the type of equation of state used (*neos,* **namelist** parameter).

For *neos=0* (Jackett and McDougall [1995] equation of state), the McDougall [1987] polynomial expression is used with the pressure in decibar approximated by the depth in meters:

$$N^2 = g\,\beta(\overline{T}^{k+1/2}, S', z_w)/e_{3w} \left(\alpha/\beta(\overline{T}^{k+1/2}, S', z_w) \quad \delta_{k+1/2}[T] - \delta_{k+1/2}[S] \right) \qquad \text{(III.3.2)}$$

where T is the *potential* temperature, S' a salinity anomaly ($S' = \overline{S}^{k+1/2} - 35.$), z_w the w-point depth (the gridpoint at which N is computed), and α (β) the thermal (haline) expansion coefficient. Both α and β depend on *potential* temperature, salinity which are averaged at w-points prior to the computation.

When a linear equation of state is used (*neos=1* or *2,* **namelist** parameter), (III.3.2) reduces to:

$$N^2 = \frac{g}{e_{3w}} \left(\beta\, \delta_{k+1/2}[S] - \alpha\, \delta_{k+1/2}[T] \right) \qquad \text{(III.3.3)}$$

where α and β are the constant coefficients used to defined the linear equation of state (III.3.1).

III.3-c Specific Heat (*rcp*, **namelist** parameter)

The specific heat, C_p, is a function of temperature, salinity and pressure [UNESCO, 1983]. It is only used in the model to convert surface heat fluxes into surface temperature increase, thus the pressure dependence is neglected. The dependence on T and S is weak. For example, with $S = 35\ psu$, C_p increases from *3989* to *4002* when T varies from *-2 °C* to *31 °C*. Therefore, C_p has been chosen as a constant: $C_p = 4.\,10^{3}\ J\ kg^{-1} \circ K^{-1}$ and is set as *rcp* (**namelist** parameter).

III.3-d Freezing Point of Seawater

The freezing point of seawater is a function of salinity and pressure [UNESCO, 1983]:

$$T_f(S,p) = -0.0575\ S + 1.710523\ 10^{-3}\ S^{2/3} - 2.154996\ 10^{-4}\ S^2 - 7.53\ 10^{-3}\ p \qquad \text{(III.3.3)}$$

(III.3.3) is only used to compute the potential freezing point of sea water (i.e. referenced to the surface *p=0*), thus the last term in (III.3.3) has been dropped. The freezing point is introduced in the model as a statement function in the *stafun.h* include file.

References

(see OPA-Bibliography when the year is in **bold**)

Bryden, H., 1973. *Deep-Sea Res., 20*, 401-408.

Jackett, D. R., and T. J. McDougall, 1995 : Minimal adjustment of hydrographic data to achieve static stability. *J. Atmos. Ocean. Tech.*, 12, 381-389.

McDougall, T. J., 1987 : Neutral surfaces. *J. Phys. Oceanogr., 17*, 1950-1964.

UNESCO, 1983: Algorithms for computation of fundamental property of sea water. *UNESCO Techn. Paper in Mar. Sci.*, 44, Unesco, 53pp.

III.4 AIR-SEA BOUNDARY CONDITIONS

III.4-a Momentum Fluxes

(default option or **key_tau** or **key_coupled** defined)

The surface boundary condition on momentum is given by the stress exerted by the wind. At the surface ($z = 0$), the momentum fluxes are prescribed as the boundary condition on the vertical turbulent momentum fluxes (see routine *dynzdf.F*),

$$\left(\frac{A^{vm}}{e_3} \frac{\partial \mathbf{U}_h}{\partial k} \right)\Big|_{z=1} = \frac{\left(\tau_u , \tau_v \right)}{\rho_o} \qquad \text{(III.4.1)}$$

where (τ_u, τ_v) are the two components of the wind stress vector in the (i, j) coordinate system.

The wind stress field is defined and updated at each time step in routine *tau.F* which is called inside the time loop by *step.F* routine. It can be prescribed analytically (default option), or by an atmospheric model through the OASIS coupler [Terray 1994] (**key_coupled** defined), or read in *numtau* file (**key_tau** defined). In the latter case, an off-line program, OPADTA, is available to interpolate either climatological or atmospheric model output wind stress field onto the ocean mesh (see § IV.3-b).

III.4-b Heat and Fresh Water Fluxes

(default option or **key_flx** or **key_coupled** defined)

At the surface ($z = 0$), the heat and fresh water fluxes are prescribed as boundary conditions on the vertical turbulent fluxes on T and S (see routine *trazdf.F*),

$$\left(\frac{A^{vT}}{e_3} \frac{\partial T}{\partial k} \right)\Big|_{z=0} = \frac{Q}{\rho_o C_p} \ , \ \left(\frac{A^{vT}}{e_3} \frac{\partial S}{\partial k} \right)\Big|_{z=0} = EMP\, S|_{z=0} \qquad \text{(III.4.2)}$$

where EMP is the evaporation minus precipitation minus river runoff plus the rate of change of the sea ice thickness budget, $S|_{z=0}$ is the sea surface salinity and Q is the non-penetrative part of the net surface heat flux, Q_t. (see § III.4-c). Note that even if the budget of EMP is zero in average over the whole ocean domain, the associated salt flux is not, as sea-surface salinity and EMP are usually highly correlated.

The Q and EMP fields are defined and updated in routine *flx.F* which, like *tau.F*, is called at each time step at the beginning of the *step.F* routine. They can be prescribed analytically (default option), or by an atmospheric model through the OASIS coupler [Terray 1994] (**key_coupled** defined), or read in from the *numflx* file (**key_flx** defined). In the latter case, an off-line program, OPADTA, is available to linearly interpolate either climatological or model output Q and EMP fields onto the model mesh (see § IV.3-b).

In forced mode (default option or **key_flx** defined), a feedback term *must* be added to the specified surface heat flux Q_o [Madec and Delecluse 1997]:

$$Q = Q_o + dQ/dT \, (T - SST) \qquad \text{(III.4.3)}$$

where SST is a sea surface temperature field (observed or climatological), T is the model surface layer temperature and dQ/dT is a negative feedback coefficient usually taken equal to $-40.\ W/m^2/°K$. For a 50 m mixed-layer depth, this value corresponds to a relaxation time scale of two months. This term ensures that if T perfectly fits SST then Q is equal to Q_o. In the fresh water budget, a feedback term can also be added:

$$EMP = EMP_o + \gamma_s^{-1} (S - SSS)/S \qquad \text{(III.4.4)}$$

where EMP_O is a net surface fresh water flux (observed, climatological or atmospheric model product), SSS is usually a time interpolation of the seasonal-mean sea surface salinity of Levitus [1982], S is the model surface layer salinity and γ_S is a negative feedback coefficient which is chosen so that the associated time scale is the same on T and S. Unlike heat flux, there is no physical justification for the feedback term in (III.4.4) as the atmosphere does not care about ocean surface salinity.

III.4-c Penetrative Solar Radiation

(**key_flxqsr** defined)

When the penetrative solar radiation option is used (**key_flxqsr** defined), the solar radiation penetrates the top few meters of the ocean, otherwise all the heat flux is absorbed in the first ocean level (default option). A formulation including extinction coefficients is assumed for the downward irradiance I [Paulson and Simpson, 1977] :

$$I(z) = Q_{sr} \left[R\, e^{-z/\xi_1} + (1 - R)\, e^{-z/\xi_2} \right] \qquad \text{(III.4.5)}$$

where Q_{sr} is the penetrative part of the surface heat flux, ξ_1 and ξ_2 are two extinction length scales and R determines the relative contribution of the two terms. The default values used correspond to a Type I water in Jerlov's [1968] classification: $\xi_1 = 0.35m$, $\xi_2 = 23m$ and $R = 0.58$ (*xsi1*, *xsi2* and *rabs* **namelist** parameters, respectively). A new term is thus added to the time evolution equation of temperature while the surface boundary condition of vertical diffusion is modified to take into account only the non-penetrative part of the surface heat flux:

$$\frac{\partial T}{\partial t} = \ldots + \frac{1}{\rho_o C_p \, e_3} \frac{\partial I}{\partial k} \qquad \text{(III.4.6)}$$
$$Q = Q_T - Q_{sr}$$

The additional term in (III.4.6) is discretized as follows:

$$\frac{1}{\rho_o C_p e_3}\frac{\partial I}{\partial k} \equiv \frac{1}{\rho_o C_p e_{3T}}\delta_k\left[I_w\right] \qquad \text{(III.4.7)}$$

with $I_w(k) = Q_{sr}\left[\, R e^{-z_w(i,j,k)/\xi_1} + (1-R) e^{-z_w(i,j,k)/\xi_2} \,\right]$. I_w is masked (no flux through the ocean bottom), so all the solar radiation that reaches the last ocean level is absorbed in that level. Note that in z-coordinates, the depth of T-levels depends on the single variable k. A one dimensional array of the coefficients $gdsr(k) = R e^{-z_w(k)/\xi_1} + (1-R) e^{-z_w(k)/\xi_2}$ can then be computed once and saved in central memory. Moreover *nksr*, the level at which *gdrs* becomes negligible (less than the computer precision) is computed once and the trend associated with the penetration of the solar radiation is only added until that level. The trend associated with the penetration of the solar radiation is added to the temperature trend and the surface heat flux modified in routine *traqsr.F*.

References

(see OPA-Bibliography when the year is in **bold**)

Jerlov N. G., 1968: Optical Oceanography, *Elsevier*, 194pp.

Levitus S., 1982: Climatological Atlas of the world ocean. NOAA *professional paper* 13, US Dept. of Commerce, Rockville, MD, 174 pp.

Paulson C. A., and J. J. Simpson, 1977: Irradiance measurements in the upper ocean. *J. Phys. Oceanogr.*, 7, 952-956.

Terray, L., 1994: The OASIS coupled user guide version 1.0. *Tech. Rep.* TR/CMGC/94-33, CERFACS.

III.5 SURFACE PRESSURE GRADIENT COMPUTATION

The computation of the surface pressure gradient with a rigid lid assumption requires to compute $\partial_t \psi$, the time evolution of the barotropic streamfunction. $\partial_t \psi$ is solution of an elliptic equation (I.2.4) for which two solvers are available, a Successive-Over-Relaxation (SOR) or a preconditioned conjugate gradient (PCG) [Rahier **1987**, Madec *et al.* **1988**, Madec **1990**]. The PCG is a very efficient method for solving elliptic equations on vector computers. It is a fast and rather easy to use method, which is an attractive feature for a large number of ocean situations (variable bottom topography, complex coastal geometry, variable grid spacing, islands, open or cyclic boundaries, etc ...). It does not require the search of an optimal parameter as in the SOR method. Nevertheless, the SOR has been kept because it is a linear solver, a very useful property when using the adjoint model of OPA.

The surface pressure gradient is computed in routine *dynspg.F*. When **key_nobsf** is defined, the barotropic flow is assumed to be zero all along the simulation so that no solver is used. This rather non-physical option can be used for debugging purposes. The default option is the use of PCG or SOR depending on *nbsfs* (**namelist** parameter). At each time step the time derivative of the barotropic streamfunction is the solution of (II.2.3). Introducing the following coefficients:

$$C_{i,j}^{N-S} = e_{2v(i,j)} \Big/ \left(H_v(i,j)\, e_{1v(i,j)}\right), \quad C_{i,j}^{E-W} = e_{1u(i,j)} \Big/ \left(H_u(i,j)\, e_{2u(i,j)}\right),$$
$$\text{and } B_{i,j} = \delta_i\left(e_{2v} M_v\right) - \delta_j\left(e_{1u} M_u\right),$$

the five-point finite difference equivalent equation (II.2.3) can be rewritten as:

$$\begin{aligned} & C_{i+1,j}^{N-S}\left(\frac{\partial\psi}{\partial t}\right)_{i+1,j} + C_{i,j+1}^{E-W}\left(\frac{\partial\psi}{\partial t}\right)_{i,j+1} \\ & + C_{i,j}^{N-S}\left(\frac{\partial\psi}{\partial t}\right)_{i-1,j} + C_{i,j}^{E-W}\left(\frac{\partial\psi}{\partial t}\right)_{i,j-1} \\ & - \left(C_{i+1,j}^{N-S} + C_{i,j}^{N-S} + C_{i,j+1}^{E-W} + C_{i,j}^{E-W}\right)\left(\frac{\partial\psi}{\partial t}\right)_{i,j} = B_{i,j} \end{aligned} \qquad \text{(III.5.1)}$$

(III.5.1) is a linear symmetric system of equations. All the elements of the corresponding matrix **A** vanish except those of five diagonals. With the natural ordering of the grid points (i.e. from west to east and from south to north), the structure of **A** is block-tridiagonal with tridiagonal or diagonal blocks. **A** is a positive-definite symmetric matrix of size $(jpi\ jpj)^2$, and **B**, the right hand side of (III.5.1), is a vector.

III.5-a Successive Over Relaxation

(*nbsfs=2*, **namelist** parameter)

Let us introduce the four cardinal coefficients: $A_{i,j}^S = C_{i,j}^{N-S} / D_{i,j}$, $A_{i,j}^W = C_{i,j}^{E-W} / D_{i,j}$, $A_{i,j}^E = C_{i,j+1}^{E-W} / D_{i,j}$ and $A_{i,j}^N = C_{i+1,j}^{N-S} / D_{i,j}$, and define $\tilde{B}_{i,j} = B_{i,j} / D_{i,j}$, where $D_{i,j} = C_{i+1,j}^{N-S} + C_{i,j}^{N-S} + C_{i,j+1}^{E-W} + C_{i,j}^{E-W}$ (i.e. the diagonal of **A**). (III.5.1) can be rewritten as:

$$\begin{aligned} & A_{i,j}^N\left(\frac{\partial\psi}{\partial t}\right)_{i+1,j} + A_{i,j}^E\left(\frac{\partial\psi}{\partial t}\right)_{i,j+1} + A_{i,j}^S\left(\frac{\partial\psi}{\partial t}\right)_{i-1,j} \\ & + A_{i,j}^W\left(\frac{\partial\psi}{\partial t}\right)_{i,j-1} - \left(\frac{\partial\psi}{\partial t}\right)_{i,j} = \tilde{B}_{i,j} \end{aligned} \qquad \text{(III.5.2)}$$

The SOR method used is an iterative method. Its algorithm can be summarised as follows [see Haltiner and Williams 1980 for further discussion]:

initialisation (evaluate a first guess from former time step computations)

$$\left(\frac{\partial\psi}{\partial t}\right)^{0}_{i,j} = 2\left(\frac{\partial\psi}{\partial t}\right)^{t}_{i,j} - \left(\frac{\partial\psi}{\partial t}\right)^{t-1}_{i,j}, \qquad \text{(III.5.3)}$$

iteration *n,* from *n=0* until convergence, do :

$$R^n_{i,j} = A^N_{i,j}\left(\frac{\partial\psi}{\partial t}\right)^{n}_{i+1,j} + A^E_{i,j}\left(\frac{\partial\psi}{\partial t}\right)^{n}_{i,j+1} + A^S_{i,j}\left(\frac{\partial\psi}{\partial t}\right)^{n+1}_{i-1,j} + A^W_{i,j}\left(\frac{\partial\psi}{\partial t}\right)^{n+1}_{i,j-1} - \left(\frac{\partial\psi}{\partial t}\right)^{n}_{i,j} - \tilde{B}_{i,j} \qquad \text{(III.5.4a)}$$

$$\left(\frac{\partial\psi}{\partial t}\right)^{n+1}_{i,j} = \left(\frac{\partial\psi}{\partial t}\right)^{n}_{i,j} + \omega\, R^n_{i,j} \qquad \text{(III.5.4b)}$$

where ω satisfies $1 \le \omega \le 2$. An optimal value exists for ω which accelerates significantly the convergence, but it has to be adjusted empirically for each model domain, except for an uniform grid where an analytical expression for ω can be found [Richtmayer and Morton 1967]. This parameter is defined as *sor*, a **namelist** parameter. The convergence test is of the form:

$$\delta = \sum_{i,j} R^n_{i,j}\, R^n_{i,j} \Big/ \sum_{i,j} \tilde{B}^n_{i,j}\, \tilde{B}^n_{i,j} \le \varepsilon \qquad \text{(III.5.5)}$$

where ε is the absolute precision that is required. It is highly recommended to use a ε smaller or equal to 10^{-2} as larger values may leads to numerically induced basin scale barotropic oscillations, and to use a two or three order of magnitude smaller value in eddy resolving configuration. The precision of the solver is not only a numerical parameter, but influences the physics. Indeed, the zero change of kinetic energy due to the work of surface pressure forces in rigid-lid approximation is only achieved at the precision demanded on the solver (§ C.1-e). The precision is specified by setting *eps* (**namelist** parameter). In addition, two other tests are used to stop the iterative algorithm. They concern the number of iterations and the module of the right hand side. If the former exceeds a specified value, *nmax* (**namelist** parameter), or the latter is greater than 10^{15}, the whole model computation is stopped while the last computed time step fields are saved in the standard output file. In both cases, this usually indicates that there is something wrong in the model configuration (error in the mesh, the initial state, the input forcing, or the magnitude of the time step or of the mixing coefficients). A typical value of *nmax* is a few hundred for $\varepsilon = 10^{-2}$, increasing to a few thousand for $\varepsilon = 10^{-12}$.

The vectorization of the SOR algorithm is not straightforward. (III.5.4) contains two linear recurrences on *i* and *j* which inhibit the vectorisation (§ IV.2-a).

Therefore (III.5.4a) has been rewritten as:

$$R^n_{i,j} = A^N_{i,j}\left(\frac{\partial\psi}{\partial t}\right)^{n}_{i+1,j} + A^E_{i,j}\left(\frac{\partial\psi}{\partial t}\right)^{n}_{i,j+1} + A^S_{i,j}\left(\frac{\partial\psi}{\partial t}\right)^{n}_{i-1,j} + A^W_{i,j}\left(\frac{\partial\psi}{\partial t}\right)^{n}_{i,j-1} - \left(\frac{\partial\psi}{\partial t}\right)^{n}_{i,j} - \tilde{B}_{i,j} \qquad \text{(III.5.6a)}$$

$$R^n_{i,j} = R^n_{i,j} - \omega\, A^S_{i,j}\, R^n_{i,j-1} \qquad \text{(III.5.6b)}$$

$$R^n_{i,j} = R^n_{i,j} - \omega\, A^W_{i,j}\, R^n_{i-1,j} \qquad \text{(III.5.6c)}$$

If the three equations in (III.5.6) are solved inside the same do-loop, (III.5.4a) and (III.5.6) are strictly equivalent. In the model they are solved successively over the whole domain. The convergence is slower but (III.5.6a) and (III.5.6b) are vector loops on *i*-index (the inner loop) and (III.5.6c) is adapted to Cray vector computers by using a routine from the Cray library (namely the FOLR routine) to solve the first-order linear recurrence. The SOR method is very flexible and can be used under a wide range of conditions, including irregular boundaries, interior boundary points, etc. Proofs of convergence, etc. may be found in the standard numerical methods texts for partial equations.

III.5-b Preconditioned Conjugate Gradient

(*nbsfs=1*, **namelist** parameter)

A is a definite positive symmetric matrix, thus solving the linear system (III.5.1) is equivalent to the minimisation of a quadratic functional:

$$\mathbf{Ax} = \mathbf{b} \Leftrightarrow \mathbf{x} = inf_{\mathbf{y}}\, \phi(\mathbf{y}), \quad \phi(\mathbf{y}) = 1/2\langle \mathbf{Ay}, \mathbf{y}\rangle - \langle \mathbf{b}, \mathbf{y}\rangle$$

where $\langle\,,\,\rangle$ is the canonical dot product. The idea of the conjugate gradient method is to search the solution in the following iterative way: assuming that $\mathbf{x}^n$ is obtained, $\mathbf{x}^{n+1}$ is searched under the form $\mathbf{x}^{n+1} = \mathbf{x}^n + \alpha^n \mathbf{d}^n$ which satisfies:

$$\mathbf{x}^{n+1} = inf_{\mathbf{y}=\mathbf{x}^n+\alpha\mathbf{d}^n}\, \phi(\mathbf{y}) \;\Leftrightarrow\; \frac{d\phi}{d\alpha} = 0$$

and expressing $\phi(\mathbf{y})$ as a function of α, we obtain the value that minimises the functional:

$$\alpha^n = \langle \mathbf{r}^n, \mathbf{r}^n\rangle / \langle \mathbf{A}\,\mathbf{d}^n, \mathbf{d}^n\rangle$$

where $\mathbf{r}^n = \mathbf{b} - \mathbf{A}\,\mathbf{x}^n = \mathbf{A}(\mathbf{x} - \mathbf{x}^n)$ is the error at rank *n*. The choice of the descent vector $\mathbf{d}^n$ depends on the error: $\mathbf{d}^n = \mathbf{r}^n + \beta^n\,\mathbf{d}^{n-1}$. β^n is searched such that the descent vectors form an orthogonal base for the dot product linked to **A**. Expressing the condition $\langle \mathbf{A}\,\mathbf{d}^n, \mathbf{d}^{n-1}\rangle = 0$ the value of β^n is found: $\beta^n = \langle \mathbf{r}^n, \mathbf{r}^n\rangle / \langle \mathbf{r}^{n-1}, \mathbf{r}^{n-1}\rangle$. As a result, the errors $\mathbf{r}^n$ form an orthogonal base for the canonic dot product while the descent vectors $\mathbf{d}^n$ form an orthogonal base for the dot product linked to **A**. The resulting algorithm is thus the following one:

initialisation :

$$\mathbf{x}^o = \left(\frac{\partial \psi}{\partial t} \right)^0_{i,j} = 2 \left(\frac{\partial \psi}{\partial t} \right)^t_{i,j} - \left(\frac{\partial \psi}{\partial t} \right)^{t-1}_{i,j}$$, the initial guess

$$\mathbf{r}^0 = \mathbf{d}^0 = \mathbf{b} - \mathbf{A}\,\mathbf{x}^0$$

$$\gamma_0 = \langle \mathbf{r}^0, \mathbf{r}^0 \rangle$$

iteration *n,* from *n=0* until convergence, do :

$$\mathbf{z}^n = \mathbf{A}\mathbf{d}^n$$

$$\alpha_n = \gamma_n / \langle \mathbf{z}^n, \mathbf{d}^n \rangle$$

$$\mathbf{x}^{n+1} = \mathbf{x}^n + \alpha_n \mathbf{d}^n$$

$$\mathbf{r}^{n+1} = \mathbf{r}^n - \alpha_n \mathbf{z}^n \qquad \text{(III.5.7)}$$

$$\gamma_{n+1} = \langle \mathbf{r}^{n+1}, \mathbf{r}^{n+1} \rangle$$

$$\beta_{n+1} = \gamma_{n+1} / \gamma_n$$

$$\mathbf{d}^{n+1} = \mathbf{r}^{n+1} + \beta_{n+1} \mathbf{d}^n$$

The convergence test is:

$$\delta = \gamma_n / \langle \mathbf{b}, \mathbf{b} \rangle \leq \varepsilon \qquad \text{(III.5.7)}$$

where ε is the absolute precision that is required. As for the SOR algorithm, both the PCG algorithm and the whole model computation are stopped when the number of iteration, *nmax*, or the module of the right hand side exceeds a specified value (see § III.5-a for further discussion). The precision and the maximum number of iteration are specified by setting *eps* and *nmax* (**namelist** parameters).

It can be demonstrated that the algorithm is optimal, provides the exact solution in a number of iterations equal to the size of the matrix, and that the convergence rate is all the more fast as the matrix is closed to identity (i.e. the eigen values are closed to 1). Therefore, it is more efficient to solve a better conditioned system which has the same solution. For that purpose, we introduce a preconditioning matrix **Q** which is an approximation of **A** but much easier to invert than **A** and solve the system:

$$\mathbf{Q}^{-1}\,\mathbf{A}\,\mathbf{x} = \mathbf{Q}^{-1}\,\mathbf{b} \qquad \text{(III.5.8)}$$

The same algorithm can be used to solve (III.5.7) if instead of the canonical dot product the following one is used: $\langle \mathbf{a}, \mathbf{b} \rangle_{\mathbf{Q}} = \langle \mathbf{a}, \mathbf{Q}\,\mathbf{b} \rangle$, and if $\tilde{\mathbf{b}} = \mathbf{Q}^{-1}\mathbf{b}$ and $\tilde{\mathbf{A}} = \mathbf{Q}^{-1}\mathbf{A}$ are substituted to **b** and **A** [Madec *et al.*, 1988]. In OPA, **Q** is chosen as the diagonal of **A**, i.e. the simplest form for **Q** so that it can be easily inverted. In this case, the discrete formulation of (III.5.8) is in fact given by (III.5.2) and thus the matrix and right hand side are computed independently from the solver used.

III.5-c Boundary Conditions: Islands

(**key_islands** defined)

The boundary condition used for both solvers is that the time derivative of the barotropic streamfunction is zero along all the coastlines. When islands are present in the model domain, additional computations must be done to determine the barotropic streamfunction with the correct boundary conditions [Madec and Marti **1990**]. This is detailed below.

The model does not recognise itself the islands which must be defined using bathymetry information, i.e. *mbathy* array, equals *-1* over the first island, *-2* over the second, ... , *-N* over the N^{th} island (see § III.2-b). The model determines the position of island grid-points and defines a closed contour around each island which will be used to compute the circulation around each island. The closed contour is formed of the ocean grid-points which are the closest to the island.

First, the island barotropic streamfunctions ψ_n are computed using the PCG (they are solutions of (III.5.1) with the right-hand side equals to zero and with $\psi_n = 1$ along the island *n* and $\psi_n = 0$ along the other coastlines) (Note that specifying *1* as boundary condition on an island for ψ is equivalent to define a specific right hand side for (III.5.1)). The precision of this computation can be very high since it is done once. The absolute precision, *epsisl*, and the maximum number of iteration, *nmisl*, are the **namelist** parameters used for that computation. Their typical values are $epsisl = 10^{-10}$ and $nmisl = 4000$. Then the island matrix A is computed from (I.2.8) and reversed. At each time step, ψ_o, the solution of (I.2.4) with $\psi_o = 0$ along all coastlines, is computed using either SOR or PCG. It has to be noted that the first guess of this computation is defined as in (III.5.3) except that $\partial_t \psi_o$ is used, not $\partial_t \psi$. Indeed we are computing $\partial_t \psi_o$ which is usually far different from $\partial_t \psi$. Then, it is easy to find the time evolution of the barotropic streamfunction on each island and to deduce $\partial_t \psi$ using (I.2.9) in order to compute the surface pressure gradient. Note that the value of the barotropic streamfunction itself is also computed as the time integration of $\partial_t \psi$ for further diagnostics.

References

(see OPA-Bibliography when the year is in **bold**)

Haltiner G. J. and R. T. Williams, 1980: *Numerical prediction and dynamic meteorology*. John Wiley & Sons Eds., second edition, 477pp.

Richtmyer, R. D., and K.W. Morton, 1967: Difference methods for initial-value problems. Interscience Publisher, Second Edition, 405pp.

III.6 LATERAL PHYSICS

The lateral physics on momentum and tracer equations have been given in §I.5-b and their discrete formulation in §II.2-c. In this section we further discuss the choices that underlie each lateral physics option. Choosing one lateral physics means for the user defining, (1) the space variation of the eddy coefficients (constant, longitude/latitude, and/or depth dependent coefficients); (2) the direction along which the lateral diffusive fluxes are evaluated (model level, geopotential or isopycnal surfaces); and (3) the type of operator used (harmonic, or biharmonic operators, and for tracers only, eddy induced advection on tracers). These three aspects of the lateral diffusion are controlled through a series of combination of cpp variables given in Table III.2). The default option is an harmonic (Laplacian) operator on both dynamics and tracer, acting along model surfaces with constant coefficient over the whole domain.

III.6-a Space Variation of Lateral Eddy Coefficients (default option or **key_dynhdfcoef~d** or **key_trahdfcoef~d** defined)

Introducing a space variation in the lateral eddy coefficients used changes the model in core memory requirement, adding up to four three-dimensional arrays for geopotential or isopycnal second order operator applied

momentum lateral physics			
direction			
	default option	:	momentum diffusion along model level surfaces
or	**key_dynhdfgeop**	:	momentum diffusion along geopotential level surfaces(available in *s*-coordinates only, as in *z*-coordinates, iso-model level diffusion is geopotential diffusion)
or	**key_dynhdfiso**	:	momentum diffusion along isopycnal surfaces (available for second order operator only)
type of operator			
	default option	:	harmonic operator (second order)
or	**key_dynhdfbilap**	:	biharmonic operator (fourth order)
eddy viscosity coefficients			
	default option	:	constant coefficient settled as a namelist parameter
or	**key_dynhdfcoef1d**	:	space varying coefficient with depth
or	**key_dynhdfcoef2d**	:	space varying coefficient with latitude and longitude
or	**key_dynhdfcoef3d**	:	space varying coefficient with latitude, longitude and depth
tracer lateral physics			
direction			
	default option	:	tracer diffusion along model level surfaces
or	**key_trahdfgeop**	:	tracer diffusion along geopotential level surfaces (available in *s*-coordinates only, as in *z*-coordinates, the iso-model level diffusion is geopotential diffusion)
or	**key_trahdfiso**	:	tracer diffusion along isopycnal surfaces (available for second order operator only)
type of operator			
	default option	:	harmonic operator (second order)
or	**key_trahdfeiv**	:	eddy induced velocity parameterisation (requires that **key_trahdfiso** is defined but not **key_trahdfbilap**)
or	**key_trahdfbilap**	:	biharmonic operator (fourth order)
eddy viscosity coefficients			
	default option	:	constant coefficient settled as a namelist parameter
or	**key_trahdfcoef1d**	:	space varying coefficient with depth
or	**key_trahdfcoef2d**	:	space varying coefficient with latitude and longitude
or	**key_trahdfcoef3d**	:	space varying coefficient with latitude, longitude and depth

Table III.2 : list of available options on momentum and tracer lateral physics.

to momentum (Table III.3). Six cpp variables control the space variation of eddy coefficients: three for momentum and three for tracer (Table III.2). They allows to specify a space variation in the three space directions, in the horizontal plane, or in the vertical only. The default option is a constant value over the whole ocean on momentum and tracers which is specified through *ahm0* and *aht0* (**namelist** parameter). The number of additional arrays that have to be defined and the gridpoint position at which they are defined depend on both the space variation chosen and the type of operator used (Table III.3). The resulting eddy viscosity and diffusivity coefficients can be either single or multiple value functions. Changes in the computer code when switching from one option to another have been minimized by introducing the eddy coefficients as statement function (include file *stafun.h*). The specification of the space variation of the coefficient is settled in *inihdf.F*, or more precisely in include files *inihdf.dyn.coef~d.h* and *inihdf.tra.coef~d.h*, with *~=1*, *2* or *3*. The user have to change these include files following his/her desiderata.

A space variation in the eddy coefficient appeals several remarks:

(1) the momentum diffusive operator acting along model level surfaces is written in terms of curl and divergent components of the horizontal current (§II.2-c). Although the eddy coefficient can be set to different values in these two terms, this option is not available.

(2) with a horizontal varying viscosity, the quadratic integral constraints on enstrophy and on the square of the horizontal divergence for operators acting along model-surfaces are no more satisfied (Appendix C).

(3) for isopycnal diffusion on momentum or tracers, an additional purely horizontal background diffusion with uniform coefficient can be added by setting a non zero value of *ahmb0* or *ahtb0*, a background horizontal eddy viscosity or diffusivity coefficient (**namelist** parameters which default value are *0.*). Nevertheless, the technique used to compute the isopycnal slopes allows to get rid of such a background diffusion which introduces spurious diapycnal diffusion (see §III.6-b).

(4) when an eddy induced advection is used (**key_trahdfeiv** defined), A^{eiv}, the eddy induced coefficient has to be defined. Its space variations are controlled by the same cpp variable as for the eddy diffusivity coefficient (i.e. **key_trahdfcoef~d** defined). The specification of the space variations of A^{eiv} is settle in *inihdf.tra.coef~d.h*. The user have to change the include file following his/her desiderata. For the default option, a constant value, *aeiv* (**namelist** parameter), is used for A^{eiv}.

(5) the eddy coefficient associated to a biharmonic operator must be set to a *negative* value.

dynamics	default option (harmonic)		key_dynhdfbilap (biharmonic)	
	default option (iso-level)	key_dynhdfiso key_dynhdfgeop	default option (iso-level)	key_dynhdfgeop (iso-z)
default option	same value everywhere	same value everywhere	same value everywhere	
key_dynhdfcoef1d array(*k*)	*T*-levels	*T*-levels and *w*-levels	*T*-levels	
key_dynhdfcoef2d array(*i,j*)	*T*- and *f*-points	*u*-, *v*- and *w*-points	*u*- and *v*-points	
key_dynhdfcoef3d array(*i,j,k*)	*T*- and *f*-points	*T*-, *uw*-, *vw*- and *f*-points	*u*- and *v*-points	

tracers	default option (harmonic)		key_trahdfbilap (biharmonic)	
	default option (iso-level)	key_trahdfiso key_trahdfgeop key_trahdfeiv*	default option (iso-level)	key_trahdfgeop (iso-z)
default option	same value everywhere	same value everywhere	same value everywhere	
key_trahdfcoef1d array(*k*)	*T*-levels	*T*-levels and *w*-levels	*T*-levels	
key_trahdfcoef2d array(*i,j*)	*u*- and *v*-points	*u*-, *v*- and *w*-points	*T*-points	
key_trahdfcoef3d array(*i,j,k*)	*u*- and *v*-points	*u*-, *v*- and *w*-points	*T*-points	

* the eddy induced velocity coefficient is defined at the same gridpoint as the eddy diffusivity coefficient.

Table III.3 : Level or gridpoint position of the lateral eddy viscosity and diffusivity coefficients as a function of the lateral physics and the vertical coordinates used.

III.6-b Lateral Tracer Physics

(default option or **key_trahdfgeop** or **key_trahdfiso** and/or **key_trahdfeiv** or **key_trahdfbilap** defined)

Three lateral diffusive operators are available in OPA: harmonic or biharmonic operators, and an eddy induced advection. Their formulation depends on the direction along which the lateral diffusive fluxes are evaluated (model level, geopotential or isopycnal surfaces) and the type of vertical coordinates used (Table III.2). In all the cases, a zero eddy fluxes of tracers through closed boundaries and sea surface is applied using the mask technique (see §II.1-c).

*** harmonic diffusive operator** (2nd order) **:**

In *z*-coordinates with geopotential diffusion (default option) or in *s*-coordinates with diffusion acting along model level (default option with **key_s_coord** defined), the diffusive operator is given by (II.2.18). It is computed in *trahdf.laplacian.h* which is included in routine *trahdf.F* (see §IV.1-b).

In *s*-coordinates geopotential or isopycnal diffusion (**key_s_coord** and **key_trahdfgeop** defined or **key_trahdfiso** defined) or in *z*-coordinates with isopycnal diffusion (**key_trahdfiso** defined), the diffusive operator is given by (II.2.17). The operator involves both lateral and vertical derivatives. For numerical stability, the second order vertical derivative must be solved using the same implicit time scheme as those used in the vertical physics (see §II.3-b). Therefore, the operator computation is split into two files, *trahdf.isoycnal.h* and *trazdf.isopycnal.h*, which solved the horizontal and vertical components of the operator, respectively. These files are included in routines *trahdf.F* and *trazdf.F*, respectively (see §IV.1-b). The slopes, r_1 and r_2, between the surface along which the diffusive operator acts and the surface of computation (*z*- or *s*-surfaces) are given by:

- *Geopotential diffusion* in *s*-coordinates: r_1 and r_2 are the slopes between the geopotential and computational surfaces. Their discrete formulation is found by locally vanishing the diffusive fluxes when *T* is horizontally uniform, i.e. by replacing in (II.2.17) *T* by z_T, the depth of *T*-point, and setting zero diffusive fluxes. This leads to the following expression for the slopes:

$$
\begin{aligned}
r_{1u} &= e_{3u} \Big/ \left(e_{1u}\, \overline{\overline{e_{3w}}}^{\,i+1/2,k} \right) \delta_{i+1/2}\left[z_T \right] \approx 1/e_{1u}\ \delta_{i+1/2}\left[z_T \right] \\
r_{2v} &= e_{3v} \Big/ \left(e_{2v}\, \overline{\overline{e_{3w}}}^{\,j+1/2,k} \right) \delta_{j+1/2}\left[z_T \right] \approx 1/e_{2v}\ \delta_{j+1/2}\left[z_T \right] \\
r_{1w} &= 1/e_{1w}\ \overline{\overline{\delta_{i+1/2}\left[z_T \right]}}^{\,i,k+1/2} \\
r_{2w} &= 1/e_{2w}\ \overline{\overline{\delta_{j+1/2}\left[z_T \right]}}^{\,j,k+1/2}
\end{aligned}
\qquad \text{(III.6.1)}
$$

These slopes are computed once in *inihdf.F*.

- *isopycnal diffusion*: r_1 and r_2 are the slopes between the isopycnal and geopotential surfaces. In *z*-coordinates, their discrete formulation is found by setting that the diffusive fluxes locally vanish when diffusing ρ instead of *T*, i.e. by replacing in (II.2.17) *T* by ρ and setting zero diffusive fluxes:

$$
\begin{aligned}
r_{1u} &= e_{3u}/e_{1u} \quad \delta_{i+1/2}[\rho] \Big/ \overline{\overline{\delta_{k+1/2}[\rho]}}^{\,i+1/2,k} \\
r_{2v} &= e_{3v}/e_{2v} \quad \delta_{j+1/2}[\rho] \Big/ \overline{\overline{\delta_{k+1/2}[\rho]}}^{\,j+1/2,k} \\
r_{1w} &= e_{3w}/e_{1w} \quad \overline{\overline{\delta_{i+1/2}[\rho]}}^{\,i,k+1/2} \Big/ \delta_{k+1/2}[\rho] \\
r_{2w} &= e_{3w}/e_{2w} \quad \overline{\overline{\delta_{j+1/2}[\rho]}}^{\,j,k+1/2} \Big/ \delta_{k+1/2}[\rho]
\end{aligned}
\qquad \text{(III.6.2)}
$$

while in *s*-coordinates their discrete formulation is found the sum of the two previous ones.

In fact the isopycnal diffusion is performed along neutral surfaces, i.e. the gradient of ρ are evaluated at the same local pressure (which in decibars is approximated by the depth in meters). Thus in (III.6.2), ρ is the *in situ* density and $\delta_{k+1/2}[\rho]$ is replaced by $-\rho\, N/g$, where *N* is evaluated following McDougall [1987] (see §III.4). These slopes are computed at each time step in *hdfslp.F*.

This implementation is similar to the one proposed by Cox [1987], except for the background horizontal diffusion. Indeed, the Cox implementation of isopycnal diffusion in GFDL-type models requires a minimum background horizontal diffusion for numerical stability reasons. To overcome this problem, several techniques have been proposed in which the numerical schemes of the OGCM are modified [Weaver and Eby 1997, Griffies *et al.* 1998]. Here, another strategy has been chosen [lazar *et al.* **1999**, Guilyardi *et al.* **1999**]: a local filtering of the isopycnal slopes (made on 9 grid-points) prevent the development of grid point noise generated by the isopycnal diffusive operator (Fig. III.5). This allows an isopycnal diffusion scheme without additional background horizontal mixing (thus referred as "pure" isopycnal mixing). This technique can be viewed as a diffusive operator that acts along large scale (2 Δx) isopycnal surfaces. The diapycnal diffusion required for numerical stability is thus minimized and its net effect on the flow

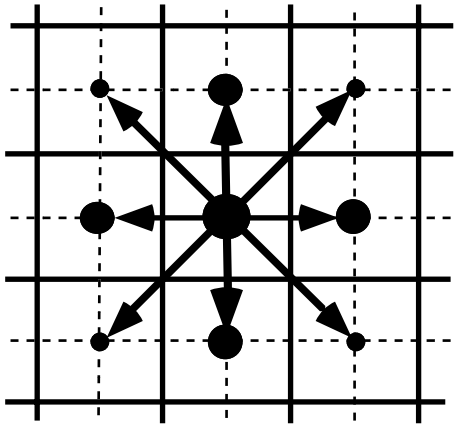


Figure III.5 : averaging procedure for isopycnal slope computation

is quite small when compared to a horizontal background mixing.

In addition and also for numerical stability reasons, the slopes are bounded by *1/100* everywhere. This limit linearly decreasing to zero between *70* meters depth and the surface (the fact that the eddies "feel" the surface motivates this flattening of isopycnals near the surface).

*** Biharmonic diffusive operator** (4th order) **:**

The biharmonic diffusive operator (**key_trahdfbilap** defined) is only available for diffusion along model level or geopotential surfaces, not along isopycnal surfaces (i.e. **key_trahdfiso** must not be defined). This restriction is due to the fact that an implicit time scheme for the high order vertical derivatives is not yet available.

In *z*-coordinates with geopotential diffusion or diffusion acting along model level in *s*-coordinates (default option with **key_s_coord** defined), the biharmonic diffusive operator is obtained by applying (II.2.18) twice. It is computed in *trahdf.bilaplacian.h* which is included in routine *trahdf.F* (see §IV.1-b).

In *s*-coordinates geopotential (**key_s_coord** and **key_trahdfgeop** defined), the biharmonic diffusive operator is obtained by applying (II.2.17) twice. All the terms of the operator are computed in *trahdf.bilapgeopot.h* which is included in routine *trahdf.F* (see §IV.1-b). The slopes used are those between geopotential and *s*-surfaces (i.e. Equations (III.6.1)).

Note that for biharmonic operator an additional boundary condition is required on the third derivative normal to the solid boundaries: it is set to zero using the mask technique.

*** Eddy induced advection :**

When Gent and McWilliams [1990] diffusion is used (**key_trahdfeiv** defined), an eddy induced tracer advection term is added, the formulation of which depends on the slopes of isopycnal surfaces. These slopes are referenced to the geopotential surfaces, i.e. (III.6.3) is used in *z*-coordinates, and (III.6.3) plus (III.6.2) for *s*-coordinates. The advection scheme is the same as those used for the true velocity (§ II.2-b). The eddy induced velocity is given by:

$$
\begin{aligned}
u^* &= \frac{1}{e_{2u}\, e_{3u}} \delta_k \left[e_{2u}\, A^{eiv}_{uw}\, \overline{r_{1w}}^{\,i+1/2} \right] \\
v^* &= \frac{1}{e_{1u}\, e_{3v}} \delta_k \left[e_{1v}\, A^{eiv}_{vw}\, \overline{r_{2w}}^{\,j+1/2} \right] \\
w^* &= \frac{1}{e_{1w}\, e_{2w}} \left\{ \delta_i \left[e_{2u}\, A^{eiv}_{uw}\, \overline{r_{1w}}^{\,i+1/2} \right] + \delta_j \left[e_{1v}\, A^{eiv}_{vw}\, \overline{r_{2w}}^{\,j+1/2} \right] \right\}
\end{aligned}
\qquad \text{(III.6.3)}
$$

where A^{eiv} is the eddy induced velocity coefficient set through *aeiv*, a **namelist** parameter. At surface, lateral and bottom boundaries, the eddy induced velocity and thus the advective fluxes of heat and salt are set to zero. The eddy induced advection is computed in the same two include files as the isopycnal operator, as both have to be used simultaneously.

III.6-c Lateral Physics on Momentum

(default option or **key_dynhdfgeop** or **key_dynhdfiso** or **key_dynhdfbilap** defined)

The discrete form of the lateral dynamic diffusive operator is given by (II.2.19), (II.2.20) or (II.2.21) depending on the use of *z*- or *s*-coordinates and of the direction along which the diffusion acts. It is written in terms of curl and divergence components of the horizontal current when the direction is along model levels and as an operator applied independently on the two components of the horizontal velocity for isopycnal diffusion or for geopotential diffusion in *s*-coordinates (see §II.2-c).

*** harmonic viscous operator** (2nd order) **:**

In *z*-coordinates with geopotential diffusion (default option), the diffusive operator is given by (II.2.21). It is computed in *dynhdf.laplacian.h* which is included in routine *dynhdf.F* (see §IV.1-b).

In *s*-coordinates geopotential or isopycnal diffusion (**key_s_coord** and default option for momentum diffusion or **key_dynhdfiso** defined) or *z*-coordinates with isopycnal diffusion (**key_dynhdfiso** defined), the diffusive operator is similar to (II.2.19) applied to each component of the velocity (cf. Eqs. (II.2.17)). The slopes between the surface along which the diffusive operator acts and the surface of computation (*z*- or *s*-surfaces) are defined at *T*-, *f*-, and *uw*-points for the *u*-component, and *f*- *T*-, *vw*-points for the *v*-component. They are computed as follows from the slopes used for tracer diffusion, i.e. (III.6.1) and (III.6.2) :

$$
\begin{aligned}
r_{1T} &= \overline{r_{1u}}^{\,i} & \qquad r_{1f} &= \overline{r_{1u}}^{\,i+1/2} \\
r_{2f} &= \overline{r_{2v}}^{\,j+1/2} & \qquad r_{2T} &= \overline{r_{2v}}^{\,j} \\
r_{1uw} &= \overline{r_{1w}}^{\,i+1/2} & \text{and} \qquad r_{1vw} &= \overline{r_{1w}}^{\,j+1/2} \\
r_{2uw} &= \overline{r_{2w}}^{\,j+1/2} & \qquad r_{2vw} &= \overline{r_{2w}}^{\,j+1/2}
\end{aligned}
\qquad \text{(III.6.4)}
$$

The major issue remains in the specification of the boundary conditions. The choice made consists in keeping the same boundary conditions as for lateral diffusion along model level surfaces, i.e. using the shear computed along the model levels and with no additional friction at the ocean bottom (see §II.1-c and §III.9).

*** Biharmonic viscous operator** (4th order)**:**

The biharmonic diffusive operator **key_dynhdfbilap** defined) is only available for diffusion along model level or geopotential surfaces, not along isopycnal surfaces (i.e. **key_dynhdfiso** must not be defined). This restriction is due to the fact that an implicit time scheme for the high order vertical derivatives is not yet available.

In *z*-coordinates with geopotential diffusion or diffusion acting along model level in *s*-coordinates (default option with **key_s_coord** defined), the biharmonic diffusive operator is obtained by applying (II.2.19) or (II.2.20) twice. It is computed in

dynhdf.bilaplacian.h which is included in routine *dynhdf.F* (see §IV.1-b).

In *s*-coordinates geopotential (**key_s_coord** and **key_dynhdfgeop** defined), the biharmonic diffusive operator is obtained by applying (II.2.21) twice. All the terms of the operator are computed in *dynhdf.bilap-geopot.h* which is included in routine *dynhdf.F* (see §IV.1-b). The slopes used are those between geopotential and *s*-surfaces (i.e. (III.6.1) averaged following (III.6.4)).

Note that for biharmonic operator an additional boundary condition is required on the third order derivatives of velocity fields at solid boundaries. The model currently has this built into the code where the biharmonic terms are computed. The high order boundary condition used is analogous to those used for the second order operator (see §II.1-c).

References
(see OPA-Bibliography when the year is in **bold**)

Cox, M. D., 1987: Isopycnal diffusion in a z-coordinate ocean model. *Ocean Modelling*, 74, 1-9.

Gent, P. R., and J. C. McWilliams, 1990: Isopycnal mixing in ocean circulation models. *J. Phys. Oceanogr., Notes and Correspondence, 20*, 150-155.

Griffies, S. M., A. Gnadadeskian, R. C. Pacanovski, V. Larichev, J. K. Dukowitcz, and R. D. Smith, 1998 : Isoneutral diffusion in a z-coordinate ocean model. *J. Phys. Oceanogr.*, in press.

McDougall, T. J., 1987: Neutral surfaces. *J. Phys. Oceanogr., 17*, 1950-1964.

Weaver, A. J., and M. Eby, 1997: On the numerical implementation of advection schemes for use in conjuction with various mixing parameterizations in the GFDL ocean model. *J. Phys. Oceanogr., 27,* 369-377.

III.7 VERTICAL PHYSICS

The discrete form of the ocean subgrid scale physics have been presented in §II.2-c. At the surface and bottom boundaries, the turbulent fluxes of momentum, heat and salt have to be defined. At the surface they are prescribed from the surface forcing (see §III.4), while at the bottom they are set to zero for heat and salt, and specified through a bottom friction parameterization for momentum (see §III.9). When the ocean is coupled to a sea-ice model, the temperature surface boundary condition switches from a Neuman to a Dirichlet boundary condition: the surface heat flux is no more specified, but diagnosed from the setting of the freezing temperature at the first ocean level.

In this section we briefly discuss the various choices offered to compute the vertical eddy coefficients, A_u^{vm}, A_v^{vm} and A^{vT}, defined at *uw*-, *vw*- and *w*-points, respectively (see §III.1). These coefficients can be assumed to be either constant, or a function of the local Richardson number, or computed from a 1.5 turbulent closure model. The computation of these coefficients is initialized in *inizdf.F* and performed in *zdfric.F* or *zdftke.F* . The trends due to the vertical momentum and tracer diffusion, including the surface forcing, are computed and added to the general trend in *dynzdf.F* and *trazdf.F*, respectively. These trends can be computed using either a forward time scheme (cpp variable **key_zdfexplicit**) or a backward time scheme (default option) depending on the magnitude of the mixing coefficients used, and thus of the formulation used (see §II.4).

III.7-a Constant (key_zdfconstant **defined)**

When the **key_zdfconstant** cpp variable is defined, the momentum and tracer vertical eddy coefficients are set to constant values over the whole ocean. This is the crudest way to define the vertical ocean physics. It is recommended to use this option only in process studies, not in basin scale simulation. Typical values used in this case are:

$$A_u^{vm} = A_u^{vm} = 10^{-4}\, m^2 s^{-1}$$

$$A^{vT} = 10^{-5}\, m^2 s^{-1}$$

These values are set through $avm0$ and $avt0$ (**namelist** parameters). In any case, do not use values smaller that those associated to the molecular viscosity and diffusivity, that is $10^{-6} m^2 s^{-1}$ for momentum, $10^{-7} m^2 s^{-1}$ for temperature and $10^{-9} m^2 s^{-1}$ for salinity.

III.7-b Richardson Number Dependent (**key_zdfrichardson** defined)

When the **key_zdfrichardson** cpp variable is defined, a local Richardson number dependent formulation of the vertical momentum and tracer eddy coefficients is set. The vertical mixing coefficients are diagnosed from the large scale variables computed by the model (order 0.5 closure scheme). *In situ* measurements allow to link vertical turbulent activity to large scale ocean structures. The hypothesis of a mixing mainly maintained by the growth of Kelvin-Helmholtz like instabilities leads to a dependency between the vertical turbulent eddy coefficients and the local stratification and vertical shear. Following Pacanowski and Philander [1981], the following formulation has been implemented:

$$A^{vT} = \frac{A_{ric}^{vT}}{\left(1 + a\, Ri\right)^n} + A_b^{vT} \;,\; A^{vm} = \frac{A^{vT}}{\left(1 + a\, Ri\right)} + A_b^{vm} \quad \text{(III.7.1)}$$

where $Ri = N^2 / \left(\partial \mathbf{U}_h / \partial z \right)^2$ is the local Richardson number, N is the local brunt-Vaisälä frequency (see §III.3-b), A_b^{vT}, A_b^{vm} are the constant background values set through *avm0* and *avt0* (**namelist** parameter), and $A_{ric}^{vT} = 10^{-4} m^2 s^{-1}$ is the maximum value that can be reached by the coefficient when $Ri \le 0$, $a = 5$ and $n = 2$. The last three variables can be modified by setting *avmri*, *alp* and *nric* (**namelist** parameter).

III.7-c 1.5 Turbulent Closure Scheme (**key_zdftke** defined)

The vertical eddy viscosity and diffusivity coefficients are computed from a 1.5 turbulent closure model based on a prognostic equation for $\bar{e}$, the turbulent kinetic energy, and a closure assumption for the turbulent length scales. This turbulent closure model has been developed by Bougeault and Lacarrère [1989] in atmospheric cases, adapted by Gaspar *et al.* [1990] for oceanic cases and embedded in OPA by Blanke and Delecluse [**1993**] for equatorial Atlantic simulations. The time evolution of $\bar{e}$ is the result* of the production of $\bar{e}$ through vertical shear, its destruction through stratification, its vertical diffusion and its dissipation of Kolmogorov' type [1942]:

$$\frac{\partial \bar{e}}{\partial t} = \frac{A^{vm}}{e_3} \left[\left(\frac{\partial u}{\partial k} \right)^2 + \left(\frac{\partial v}{\partial k} \right)^2 \right] - A^{vT} N^2 + \frac{1}{e_3} \frac{\partial}{\partial k} \left[\frac{A^{vm}}{e_3} \frac{\partial \bar{e}}{\partial k} \right] - c_\varepsilon \frac{\bar{e}^{3/2}}{l_\varepsilon} \tag{III.7.2}$$

$$A^{vm} = C_k \, l_k \sqrt{\bar{e}} \quad , \quad A^{vT} = A^{vm} / P_{rt} \tag{III.7.3}$$

where N designates the local brunt-Vaisälä frequency (see §III.3-b), l_ε and l_κ are the dissipation and mixing turbulent length scales, P_{rt} is the Prandtl number. The constants $C_k = 0.1$ and $C_\varepsilon = \sqrt{2}/2$ are designed to deal with vertical mixing at any depth [Gaspar *et al.* 1990]. They are set through **namelist** parameter *ediff* and *ediss*. P_{rt} can be set to unity or, following Blanke and Delecluse [**1993**], be a function of the local Richardson number, R_i: P_{rt} is equal to *1* for $R_i \le 0.2$ and equal to *10* for $R_i \ge 2$, with a linear transition in-between. In addition, a Shapiro filter can be optionally applied to R_i in the horizontal. The choice of P_{rt} is controlled by *npdl* (**namelist** parameter).

For computational efficiency, the original formulation of the turbulent length scales proposed by Gaspar *et al.* [1990] has been simplified. Three formulations are proposed, the choice of which is controlled by *nmxl* (**namelist** parameter). The first two are based on the following first order approximation [Blanke and Delecluse **1993**]:

$$l_k = l_\varepsilon = \sqrt{2\bar{e}} \; N \tag{III.7.4}$$

which is obtained in a stable stratified region with constant values of the brunt-Vaisälä frequency. The resulting turbulent length scale is bounded by the distance to the surface or to the bottom (*nmxl =0*) or by the local vertical scale factor (*nmxl =1*). Blanke and Delecluse [**1993**] notice that this simplification has two major drawbacks: it has no sense for local unstable stratification and the computation no longer uses the whole information contained in the vertical density profile. With *nmxl =3*, these drawbacks are overcome by the adjunction of an additional hypothesis on the vertical gradient of the computed length scale:

$$\frac{1}{e_3} \left| \frac{\partial l}{\partial k} \right| \le 1 \tag{III.7.5}$$

(III.7.5) means that the vertical variations of the length scale cannot be larger than the variations of depth. It provides a better approximation of the Gaspar *et al.* [1990] formulation while being much less time consuming. In particular, it allows the length scale to be limited not only by the distance to the surface or to the ocean bottom but also by the distance to a strongly stratified portion of the water column such as the thermocline.

At the sea surface the value of $\bar{e}$ is prescribed from the wind stress field: $\bar{e} = ebb \, |\tau| / \rho_o$ ($ebb = 3.75$ by default) with a minimal threshold of $emin0 = 10^{-4} \; m^2 \, s^{-2}$ (**namelist** parameters). Its bottom value is assumed to be equal to the value of the level just above. The time integration of the $\bar{e}$ equation may formally lead to negative values because the numerical scheme does not ensure the positivity. To overcome this problem, a cut-off in the minimum value of $\bar{e}$ is used. Following Gaspar *et al.* [1990], the cut-off value is set to $\sqrt{2}/2 \; 10^{-6} \; m^2 \, s^{-2}$. This allows the subsequent formulations to match Gargett's [1984] one for the diffusion in the thermocline and deep ocean ($A^{vT} = 10^{-3}/N$). In addition, a cut-off is applied on A^{vm} and A^{vT} to avoid numerical instabilities associated with too weak vertical diffusion. They must be specified at least larger than the molecular values, and are set through *avm0* and *avt0* (**namelist** parameters).

* *Following Blanke and Delecluse [**1993**], the advective terms have been neglected. This assumption may be relaxed for eddy resolving simulations.*

References

(see OPA-Bibliography when the year is in **bold**)

Bougeault P., and P. Lacarrère, 1989: Parameterization of orography-induced turbulence in a mesobeta-scale model. *Mon. Wea. Rev.*, 117, 1872-1890.

Gargett A. E., 1984: Vertical eddy diffusivity in the ocean interior. *J. Mar. Res.*, *42*, 359-393.

Gaspar P., Y. Grégoris, and J. M. Lefevre, 1990: A simple eddy-kinetic-energy model for simulations of the ocean vertical mixing: tests at station Papa and Long-Term Upper Ocean Study Site site. *J. Geophys. Res.*, 95, 16,179-16,193.

Kolmogorov A. N., 1942: The equation of turbulent motion in an incompressible fluid. *Izv. Akad. Nauk SSSR*, *Ser. Fiz.*, 6, 56-58.

Pacanowski R. C., and S. G. H. Philander, 1981: Parameterization of vertical mixing in numerical models of tropical oceans. *J. Phys. Oceanogr.*, 11, 1443-1451.

III.8 CONVECTION

Static instabilities (i.e. light potential densities under heavy ones) may occur at particular ocean grid points. In nature, convective processes quickly re-establish the static stability of the water column. These processes have been removed from the model via the hydrostatic assumption: they must be parameterized. Three parameterisations are available to deal with convective processes: either a non-penetrative convective adjustment or an enhanced vertical diffusion, or/and the use of a turbulent closure scheme.

III.8-a Non-Penetrative Convective Adjustment (key_convnpc **defined)**

The non-penetrative convective adjustment algorithm is used when the **key_convnpc** cpp variable is defined. It is applied at each time step and mixes downwards instantaneously the statically unstable portion of the water column, but only until the density structure becomes neutrally stable (i.e. until the mixed portion of the water column has *exactly* the density of the water just below) [Madec *et al.* **1991a**, **1991b**]. This algorithm is an iterative process used in the following way (Fig. III.5): going from the top of the ocean towards the bottom, the first instability is searched. Assume in the following that the instability is located between levels k and $k+1$. The two levels are vertically mixed, for potential temperature and salinity, conserving the heat and salt contents of the water column. The new density is then computed by a linear approximation. If the new density profile is still unstable between levels $k+1$ and $k+2$, levels k, $k+1$ and $k+2$ are then mixed. This process is repeated until stability is established below the level k (the mixing process can go down to the ocean bottom). The algorithm is repeated to check if the density profile between level $k-1$ and k is unstable and/or if there is no deeper instability.

This algorithm is slightly different from mixing to by two statically unstable levels. The latter procedure cannot converge with a finite number of iterations for some vertical profiles while the algorithm used in OPA converges for any profile in a number of iterations less than the number of vertical levels. This property is of paramount importance as pointed out by Killworth [1989]: it avoids the existence of permanent and unrealistic static instabilities at the sea surface. This non-penetrative convective algorithm has been proved successful in studying the deep water formation in the north-western Mediterranean Sea [Madec *et al.* **1991a**, **1991b**, **1991c**].

Note that in this algorithm the potential density referenced to the sea surface is used to check whether the density profile is stable or not. Moreover, the mixing in potential density is assumed to be linear. This assures the convergence of the algorithm even when the equation of state is non-linear. Small static instabilities can thus persist due to cabbeling: they will be treated at the next time step. Moreover, temperature and salinity, and thus density, are mixed, but the corresponding velocity fields remain unchanged. When using a Richardson dependent eddy viscosity, the mixing of momentum is done through the vertical diffusion: after a static adjustment, the Richardson number is zero and thus the eddy viscosity coefficient is at a maximum. When this algorithm is used with constant vertical eddy viscosity, spurious solution can occur as the vertical momentum diffusion remains small even after a static adjustment. In that latter case, we recommend to add momentum mixing in a manner that mimics the mixing in temperature and salinity [Speich, **1992**, Speich *et al.* **1996**], or to choose the enhanced vertical diffusion parameterisation (see next sub-section).

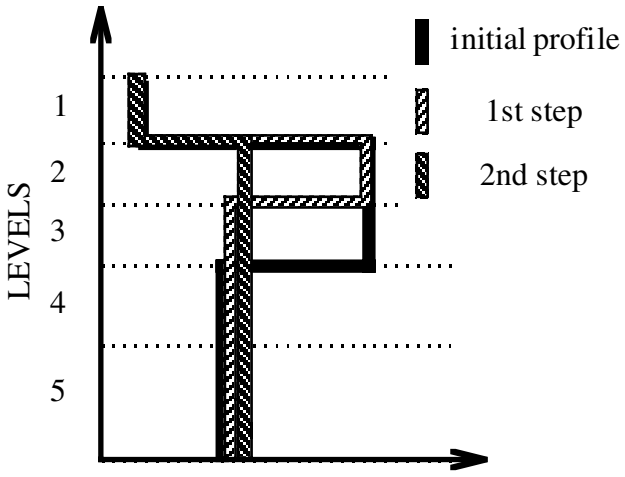


Figure III.5 : Example of an unstable density profile treated by the non penetrative convective adjustment algorithm. 1st step: the initial profile is checked from the surface to the bottom. It is found to be unstable between levels 3 and 4. They are mixed. The resulting ρ is still larger than ρ(5): levels 3 to 5 are mixed. The resulting ρ is still larger than ρ(6): levels 3 to 6 are mixed. The 1st step ends since the density profile is then stable below the level 3. 2nd step: the new ρ profile is checked following the same procedure as in 1st step: levels 2 to 5 are mixed. The new density profile is checked. It is found stable: end of algorithm.

III.8-b Enhanced Vertical Diffusion
(**key_convevd** defined)

The enhanced vertical diffusion parameterization is used when the **key_convevd** cpp variable is defined. In this case, the vertical eddy coefficients on both momentum and tracers are assigned to be very large (a typical value is $1\,m^2s^{-1}$) in regions where the stratification is unstable (i.e. when the Brunt-Vaisälä frequency is negative) [Lazar **1997**, Lazar *et al.* **1998**].

In practice, when $N^2 \leq 10^{-15}$, A_T^{vT} and the four neighbouring A_u^{vm} and A_v^{vm} are set to a large value, *avevd* (**namelist** parameter). The default value for *avevd* is $1\,m^2s^{-1}$ but a ten times larger value can be used. This parameterization of convective processes is less time consuming than the convective adjustment algorithm presented above when mixing both tracers and momentum in case of static instabilities. It requires the use of an implicit time stepping on vertical diffusion terms (i.e. default option , not **key_zdfexplicit** defined). This is automatically checked in *parctl.F* routine.

III.8-c Turbulent Closure Scheme
(**key_zdftke** defined)

The turbulent closure scheme presented in §III.7-c allows to deal with statically unstable density profile. It is used when the **key_zdftke** cpp variable is defined. In such a case, the term of destruction of turbulent kinetic energy through stratification in (III.7.2) becomes a source term as N^2 is negative. It results large values of both A_T^{vT} and the four neighbouring A_u^{vm} and A_v^{vm} (up to $1\,m\,s^{-1}$) which are able to restore the static stability of the water column in a way similar to that of the enhanced vertical diffusion parameterization (§III.8-b). Nevertheless, the eddy coefficients computed by the turbulent scheme do usually not exceed $10^{-2}\,m\,s^{-1}$ in the vicinity of the sea surface (first ocean layer) due to the bound of the turbulent length scale by the distance to the sea surface (see §III.7-c). It can thus be useful to combine the enhanced vertical diffusion with the turbulent closure, i.e. defining **key_convevd** and **key_zdftke** cpp variables all together.

References
(see OPA-Bibliography when the year is in **bold**)

Killworth, P. D., 1989 : On the parameterization of deep convection in ocean models. In *Parameterization of small-scale processes*. Proc. Hawaiian winter workshop, University of Hawaii at Manoa, January 17-20, 59-74.

III.9 BOTTOM FRICTION

Both surface momentum flux (wind stress) and the bottom momentum flux (bottom friction) enter the equations as a condition on the vertical diffusive flux. For the bottom boundary layer, one has:

$$A^{vm}\left(\partial \mathbf{U}_h / \partial z\right) = \mathbf{F}_h, \qquad \text{(III.9.1)}$$

where $\mathbf{F}_h$ is supposed to represent the horizontal momentum flux outside the logarithmic turbulent boundary layer (thickness of the order of $1\,m$ in the ocean). How $\mathbf{F}_h$ influences the interior depends on the vertical resolution of the model near the bottom relative to the Ekman layer depth. For example, in order to obtain an Ekman layer depth $d = \sqrt{2A^{vm}/f} = 50\,m$, one needs a vertical diffusion coefficient $A^{vm} = 0.125\,m^2s^{-1}$ (for a Coriolis frequency $f = 10^{-4}\,s^{-1}$). With a background diffusion coefficient $A^{vm} = 10^{-4}\,m^2s^{-1}$, the Ekman layer depth is only $1.4\,m$. When the vertical mixing coefficient is small, using a flux condition is equivalent to entering the viscous forces (either wind stress or bottom friction) as a body force over the depth of the top or bottom model layer. To illustrate this, consider the equation for u at k, the last ocean level:

$$\frac{\partial u(k)}{\partial t} = \frac{1}{e_{3u}}\left[A^{vm}(k)\,\frac{u(k-1)-u(k)}{e_{3uw}(k-1)} - F_u\right] \approx -\frac{F_u}{e_{3u}} \qquad \text{(III.9.2)}$$

For example, if the bottom layer thickness is $200\,m$, the Ekman transport will be distributed over that depth. On the other hand, if the vertical resolution is high ($1\,m$ or less) and a turbulent closure model is used, the turbulent Ekman layer will be represented explicitly by the model. However, the logarithmic layer is never represented in current primitive equation model applications: it is *necessary* to parameterize the flux $\mathbf{F}_h$. Two choices are available in OPA: a linear and a quadratic bottom friction. Note that in both cases, the rotation between the interior velocity and the bottom friction is neglected in the present release of OPA.

III.9-a Linear Bottom Friction

(**namelist** : *nbotfr = 0, = 1 or = 3*)

The linear bottom friction parameterization assumes that the bottom friction is proportional to the interior velocity (i.e. the velocity of the last model level):

$$\mathbf{F}_h = \frac{A^{vm}}{e_3} \frac{\partial \mathbf{U}_h}{\partial k} = r\, \mathbf{U}_h^b \qquad \text{(III.9.3)}$$

where $\mathbf{U}_h^b$ is the horizontal velocity vector of the bottom ocean layer and r a friction coefficient in $m\,s^{-1}$. This coefficient is generally estimated by setting a typical decay time τ in the deep ocean, $r = H/\tau$. Commonly accepted values of τ are of the order of *100* to *200* days [Weatherly 1984]. A value $\tau^{-1} = 10^{-7}\ s^{-1}$ corresponding to *115* days is usually used in quasi-geostrophic models. One may consider the linear friction as an approximation of quadratic friction, $r \approx 2\, C_D\, U_{av}$ [Gill 1982, Eq. 9.6.6]. With a drag coefficient $C_D = 0.002$, a typical value of tidal currents $U_{av} = 0.1\, m\,s^{-1}$, and assuming an ocean depth $H = 4000\, m$, the resulting friction coefficient is $r = 4\ 10^{-4}\, m\,s^{-1}$. This is the default value used in OPA. It corresponds to a decay time scale of *115* days. It can be changed by specifying *bfric1* (**namelist** parameter).

In the code, the bottom friction is specified by updating the value of the vertical eddy coefficient at the bottom level. Indeed, the discrete formulation of (III.9.3) at the last ocean *T*-level, using the fact that $\mathbf{U}_h = 0$ inside the bottom, leads to

$$\begin{aligned} A_u^{vm} &= r\, e_{3uw} \\ A_v^{vm} &= r\, e_{3vw} \end{aligned} \qquad \text{(III.9.4)}$$

Such an update is done in *zdfbfr.F* when *nbotfr=1* (**namelist** parameter) and the value of *r* used is *bfric1* (**namelist** parameter). Setting *nbotfr=3* (**namelist** parameter) is equivalent to set $r = 0$ and leads to a free-slip bottom boundary condition, while setting *nbotfr=0* (**namelist** parameter) imposes $r = 2\, A_{vb}^{\mathbf{U}}$, where $A_{vb}^{\mathbf{U}}$ is the background vertical eddy coefficient: a no-slip boundary condition is used. Note that this latter choice generally leads to an underestimation of the bottom friction: for a deepest level thickness of *200 m* and $A_{vb}^{\mathbf{U}} = 10^{-4}\, m^2 s^{-1}$, the friction coefficient is only $r = 10^{-6}\, m\,s^{-1}$.

III.9-b Non-Linear Bottom Friction

(**namelist** : *nbotfr = 2*)

The non-linear bottom friction parameterization assumes that the bottom friction is quadratic:

$$\mathbf{F}_h = \frac{A^{vm}}{e_3} \frac{\partial \mathbf{U}_h}{\partial k} = C_D \sqrt{u_b^{\,2} + v_b^{\,2} + e_b}\ \ \mathbf{U}_h^b \qquad \text{(III.9.5)}$$

with $\mathbf{U}_h^b = (u_b, v_b)$ the horizontal interior velocity (i.e. the horizontal velocity of the bottom ocean layer), C_D a drag coefficient, and e_b a bottom turbulent kinetic energy due to tides, internal waves breaking and other short time scale currents. A typical value of the drag coefficient is $C_D = 10^{-3}$.As an example, the CME experiment uses $C_D = 10^{-3}$ and $e_b = 0$, while the FRAM experiment uses $C_D = 10^{-3}$ and $e_b = 2.5\ 10^{-3} m^2 s^{-2}$. The FRAM choices have been set as default value (*bfric2* and *bfeb2*, **namelist** parameters).

As for the linear case, the bottom friction is specified in the code by updating the value of the vertical eddy coefficient at the bottom level:

$$\begin{aligned} A_u^{vm} &= C_D\, e_{3uw} \left[u^2 + \left(\overset{=i+1,j}{v} \right)^2 + e_b \right]^{1/2} \\ A_v^{vm} &= C_D\, e_{3vw} \left[\left(\overset{=i,j+1}{u} \right)^2 + v^2 + e_b \right]^{1/2} \end{aligned} \qquad \text{(III.9.6)}$$

This update is done in *zdfbfr.F*. The coefficients that control the strength of the non-linear bottom friction are initialized as **namelist** parameters: ($C_D = bfri2$, and $e_b = bfeb2$).

References

(see OPA-Bibliography when the year is in **bold**)

Gill A., 1982: Atmosphere Ocean Dynamics, International Geophysics Series, *New-York: Academic Press.*

Weatherly, G. L., 1984 : An estimate of bottom frictional dissipation by Gulf Stream fluctuations. *J. Mar. Res., 42*, 289-301.

III.10 LATERAL MODEL DOMAIN BOUNDARY CONDITIONS

The lateral ocean boundary conditions continuous to coastlines are Neumann conditions for heat and salt (no flux across boundaries) and Dirichlet conditions for momentum (from free- to strong-slip). They are handled automatically by the mask system (§II.1-c). There are some minor variations on these at the model domain boundaries. Four choices are offered: closed, cyclic east-west, symmetric across the equator, and open boundary conditions.

III.10-a Closed, Cyclic or Symmetric Conditions

(*jperio* = *0*, *1*, or *2*, model **parameter**)

The choice of closed, cyclic or symmetric model domain boundary condition is made by setting *jperio* to *0*, *1* or *2* in *parameter.h* file. Each time such a boundary condition is needed, it is set by a call to *lbc.F* or *mpplnk.F* routine depending on the architecture of the computer used (default option or **key_mpp** defined). The computation of momentum and tracer trends proceed from $i = 2$ to $i = jpi - 1$ and from $j = 2$ to $j = jpj - 1$. To choose a lateral model boundary condition is to specify the first and last rows and columns of the model variables.

- For closed boundary (*jperio=0*), solid walls are imposed at all model boundaries: first and last rows and columns are set to zero.

- For cyclic east-west boundary (*jperio=1*), first and last rows are set to zero (closed) while first column is set to the value of the before last column and last column to the value of the second one (Fig. III.6a). Whatever flows out of the eastern (western) end of the basin enters the western (eastern) end. Note that there is neither option for north-south cyclic nor doubly cyclic cases.

- For symmetric boundary condition across the equator(*jperio=2*), last rows, and first and last columns are set to zero (closed). The row of symmetry is chosen to be the *u*- and *T*-points equator line ($j = 2$, i.e. at the southern end of the domain). For arrays defined at *u*- or *T*-points, the first row is set to the value of the third row while for most of v- and f-points arrays (v, ζ, ψ, but scalar arrays such as eddy coefficients) the first row is set to minus the value of the second row (Fig. III.6-b). Note that this boundary condition is not yet available on massively parallel computer (**key_mpp** defined).

III.10-b Open Boundary Conditions

(**key_eastobc**, and/or **key_westobc**, and/or **key_northobc**, and/or **key_southobc** defined)

Not available in release 8.1, but already implemented in the 8.2 beta release for CLIPPER project.

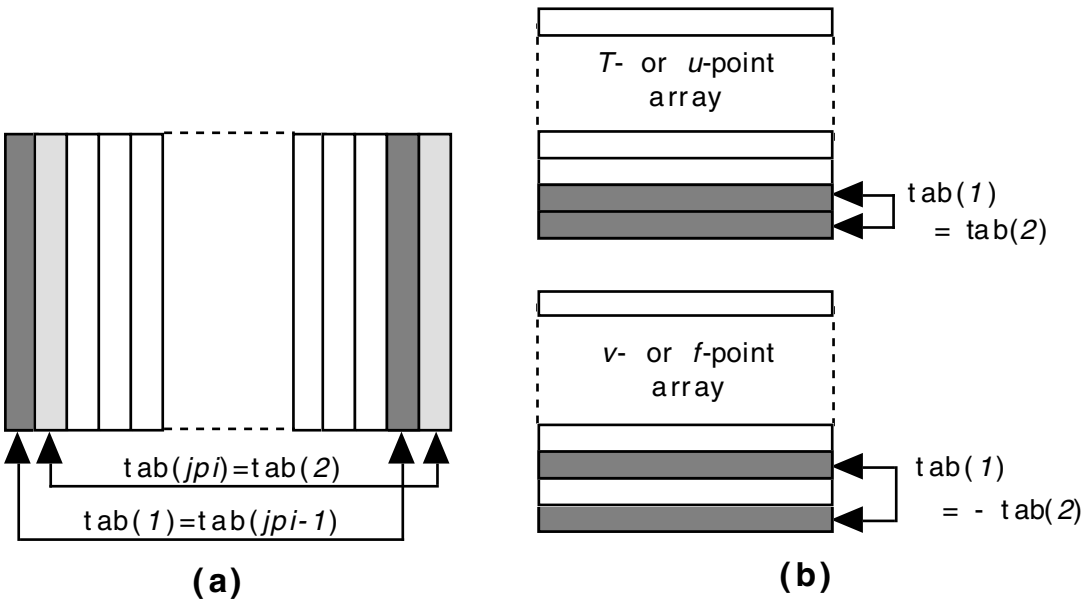


Figure III.6: setting of (a) east-west cyclic (b) symmetric across the equator boundary conditions.

III.11 ANNEXE FUNCTIONALITIES

III.11-a Internal Restoring Term on *T*-S Fields (**key_tradmp** defined)

In some applications it can be useful to add a Newtonian damping term in the temperature and salinity equations :

$$\frac{\partial T}{\partial t} = \cdots - \gamma (T - T_o) \qquad \frac{\partial S}{\partial t} = \cdots - \gamma (S - S_o) \tag{III.11.1}$$

where γ is the inverse of a time scale, and T_o and S_o are given temperature and salinity fields. The restoring term is added if the **key_tradmp** cpp variable is defined. It requires that both cpp variables **key_temdta** and **key_saldta** are defined (i.e. that T_o and S_o are read). The restoring coefficient γ is a three-dimensional array initialized by the user in *dtacof.F*. The additional damping term is added in *tradmp.F*.

The two main cases in which (III.11.1) is used are (1) the specification of the boundary conditions along artificial walls of a limited domain basin and (2) the computation of the velocity field associated with a given *T-S* field (for example to build the initial state of a prognostic simulation, or to use the resulting velocity field for a passive tracer study). In the first case, regional models usually have artificial walls instead of open boundaries. In the vicinity of these walls, γ takes large values (equivalent to a few day time scale) whereas it is zero in the interior of the model domain. The second case corresponds to the use of the robust diagnostic method [Sarmiento and Bryan 1982]. It allows to find the velocity field consistent with the model dynamics while having a *T-S* field close to a given climatology field (T_o - S_o). The time scale associated with γ is generally not a constant but spatially varying in order to respect some considerations. For example, it is usually set to zero in the mixed layer (defined either on a density or A^{vT} criterion) [Madec and Imbard **1996**] and in the equatorial region [Reverdin *et al.* **1991**, Fujio and Imasato 1991, Marti **1992**] as those two regions have a small time scale of adjustment, while smaller γ are used in the deep ocean where the typical time scale is long [Sarmiento and Bryan 1982]. In addition it is reduced (and even zero) along the western boundary to allow the model to reconstruct its own western boundary structure in equilibrium with its physics. The choice of a Newtonian damping acting in the mixed layer or not is controlled by *nmldmp* (**namelist** parameter).

The robust diagnostic method is very efficient to prevent the temperature drift in intermediate waters but it produces artificial sources of heat and salt within the ocean. It has also undesirable effects on the ocean convection. It tends to prevent deep convection and subsequent deep water formation by stabilising too much the water columns.

III.11-b Accelerating the Convergence (*nacc = 1*,**namelist** parameter)

Searching an equilibrium state with an ocean model requires very long time integration (a few thousand years for a global model). Due to the size of the time step required for numerical stability consideration (less than a few hours), this is usually quite expensive even on a super-computer and requires a large elapse time. In order overcome this problem, Bryan [1984] introduces a technique which allows to accelerate the spin up to the equilibrium. The technique basically consists in using a larger time step in the thermodynamic evolution equations than in the dynamic evolution equations. It does not affect the equilibrium solution but modifies the trajectory to reach it.

The acceleration of convergence is used when *nacc=1* (**namelist** parameter) while synchroneous time stepping is achieved when *nacc=0*. In that case, $\Delta t = rdt$ is the time step of dynamics while $\Delta \tilde{t} = rdttra$ is the tracer time step. Both are settled from **namelist** parameters. The set of prognostic equations to solve becomes :

$$\frac{\partial \mathbf{U}_h}{\partial t} \equiv \frac{\mathbf{U}_h^{\,t+1} - \mathbf{U}_h^{\,t-1}}{2\Delta t} = \ \ldots \qquad \frac{\partial T}{\partial t} \equiv \frac{T^{t+1} - T^{t-1}}{2\Delta \tilde{t}} = \ \ldots \qquad \frac{\partial S}{\partial t} \equiv \frac{S^{t+1} - S^{t-1}}{2\Delta \tilde{t}} = \ \ldots \tag{III.11.2}$$

Bryan [1984] has analysed the consequences of this distorted physics. Free waves have a slower phase speed, their meridional structure is slightly modified, and the growth rate of baroclinically unstable waves is reduced but there is a wider range of instability. This technique is efficient for searching an equilibrium state in coarse resolution models. However its application is not suitable for many oceanic problems: it cannot be used for transient or time evolving problems (in particular, it is very questionable to keep this technique when studying a mean seasonal cycle), and it cannot be used in high resolution models where baroclinically unstable processes are important. Moreover, the vertical variation of $\Delta \tilde{t}$ implies that the heat and salt contents are no more conserved due

to the vertical coupling of the ocean level through both advection and diffusion.

III.11-c Zoom Functionality

(*nizoom, njzoom*, **namelist** parameters)

The zoom option offered in OPA is a quite simple function that allows to perform a simulation over a sub-domain of an already defined configuration (i.e. without defining a new set of mesh, initial state and forcings). This option can be useful for testing the user setting of surface boundary conditions, or the initial ocean state of a huge ocean model configuration while having a small in core memory requirement. It can also be used to easily test specific physics in a sub-domain (for example, test of the coupling between sea-ice and ocean models over the Arctic or Antarctic ocean as it is set in the global ocean configuration [Madec and Imbard **1996**]). In standard, this option does not include any specific treatment for ocean boundaries of the sub-domain: they are considered as artificial vertical walls. Nevertheless, it is quite easy to add a restoring term toward a climatology in the vicinity of those boundaries (see §III.11-a).

In order to easily define a sub-domain over which the computation can be performed, the dimension of all input arrays (ocean mesh, bathymetry, forcing, initial state, ...) are defined as *jpidta*, *jpjdta* and *jpkdta* (*parameter.h*), while the computational domain is defined through *jpiglo*, *jpjglo* and *jpk* (*parameter.h*). When running the model over the whole configuration, the user set *jpiglo=jpidta* *jpjglo=jpjdta* and *jpk=jpkdta*. When running the model over a sub-domain, the user have to provide the size of the sub-domain, (*jpiglo*, *jpjglo*, *jpk*) in the *parameter.h* file, and the indices of the south western corner as *nizoom* and *njzoom* (**namelist** parameters) (Fig. III.7).

Note that a third set of dimension exist, *jpi*, *jpj* and *jpk* which is actually used to perform the computation. It is set by default to *jpi=jpjglo* and *jpj=jpjglo*, except for massively parallel computing where the computational domain is laid out on local processor memories following a 2D horizontal splitting (see §IV.2-c)

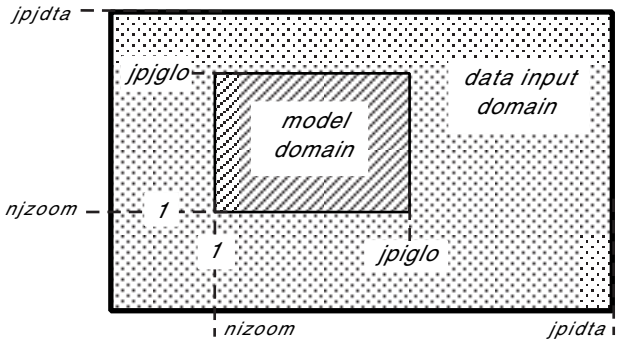


Figure III.7 : position of a model domain compared to the data input domain when the zoom functionality is used.

References

(see OPA-Bibliography when the year is in **bold**)

Bryan, K., 1984: Accelerating the convergence to equilibrium of ocean-climate models, *J. Phys. Oceanogr.*, 14, 666-673.

Fujio, S., and N. Imasato, 1991 : Diagnostic calculation for circulation and water mass movement in the deep Pacific. *J. Geophys. Res., 96*, C1, 759-774.

Sarmiento, J. L., and K. Bryan, 1982 : An ocean transport model for the north Atlantic. *J. Geophys. Res., 87, C1*, 394-408.

III.12 DIAGNOSTICS

III.12-a Standard Model Output

(default option or **key_diainstant** defined)

The model outputs are of three types: the restart file, the output listing, and the output file(s). The restart file is used internally by the code when the user wants to start the model with initial conditions defined by a previous simulation. It contains all the information that is necessary not to introduce changes in the model results (even at the computer precision) between a run performed with several restarts and the same run performed in one time. It has to be noticed that this requires that the restart file contains two consecutive time steps for all the prognostic variables, and that it is save in the same binary format as the one used by the computer to calculate (in particular, 32 bits binary IEEE format for this file must not be used). The output listing and file(s) are defined but should be checked and eventually adapted to the user's needs. The output listing is stored in the *ocean.output* file. The information are printed all the way through the code on the logical unit *numout*. To locate these prints, use the UNIX command ' grep -i numout * ' in the source code directory.

In the standard configuration, the user will find the model results in two output files for every time-step where output is demanded: a **VO** file containing all the three dimensional fields in logical unit *numwvo*, and a **SO** file containing all the two dimensional (horizontal) fields in logical unit *numwso*. These outputs are defined in the *diawri.F* routine. The standard and available on-line diagnostics are described in §III-12-c.

The default output is 32 bits binary IEEE format, compatible with the Vairmer software (see the Climate Modelling and global Change team WEB server at CERFACS: http://www.cerfacs.fr). The model's reference directory also contains a visualisation tool based on **NCAR Graphics** (http://ngwww.ucar.edu). If a user has access to the NCAR software, he or she can copy the **LODMODEL/UTILS/OPADRA** directory from the reference and, following the **README**, create the graphic outputs from the model's results.

III.12-b Tracer/Dynamics Trends (**key_diatrddyn** or **key_diatrdtra** defined)

When **key_diatrddyn** and/or **key_diatrddyn** cpp variables are defined, each trends of the dynamics and/or temperature and salinity time evolution equations are stored in three-dimensional arrays just after their computation (i.e. at the end of each *dyn···.F* and/or *tra···.F* routines). These trends are then used in diagnostic routines *diadyn.F* and *diatra.F* respectively. In standard, these routines check the basin averaged properties of the momentum and tracer equations every *ntrd* time-steps (**namelist** parameter). These routines are given as an example, they must be adapted by the user to his/her desiderata.

These two options imply the definition of several arrays in the in core memory, increasing quite sensitively the in core memory requirements (search for **key_diatrddyn** or **key_diatrdtra** in *common.h* file)

III.12-c Other Diagnostics

Aside from the standard model variables, some other diagnostics are computed on-line or can be added in the model. The available ready-to-add diagnostics can be found in the reference directory (see §IV.3) in **LODMODEL/UTILS/OPADIAGS**. Among the available diagnostics one can quote:

- the mixed layer depth (based on a density criterion) (*diamxl.F*)
- the turbocline depth (based on a turbulent mixing coefficient criterion) (*diamxl.F*)
- the depth of the 20°C isotherm (*diahth.F*)
- the depth of the thermocline (maximum of the vertical temperature gradient) (*diahth.F*)
- the meridional heat and salt transports and their decomposition (*diaznl.F*)
- the surface pressure (*diaspr.F*)

IV. COMPUTER CODE

IV.1 CODE ARCHITECTURE

The computer code has been developed as a research tool. Users are scientists who intend to produce and analyse ocean simulations in various configurations usually requiring a huge amout of computation, and possibly modify the numerical model in order to better represent the ocean physics or dynamics. Therefore, the code must be easy to understand and to modify by non specialists in computer science, although it is run on various sophisticated and powerful computers (vectorial/scalar weakly/massively parallel computers). In order to achieve this goal, the computer code has been written to be first readable and modular, and second as fast as possible on quite different computers while keeping the first principle.

IV.1-a Aims

- **readability/modularity:** The code is written in standard FORTRAN 77 with a few FORTRAN 90 extensions. Special coding tricks are avoided. It is on-line documented (about 16,300 of the 28,400 model lines are comment lines) as well as off-line documented through the present manual. Each subroutine has its own header that can be easily extracted using the standard UNIX *grep* command (see Appendix D). Details of coding rules are presented in Appendix D. The code is composed of 130 routines. Each routine is small (usually less than 200 FORTRAN lines, including comments) and well defined (i.e. dedicated to one specific function). All the FORTRAN declarations of arrays, parameters, statement functions used in more than one routine are collected in three files, *common.h*, *parameter.h* and *stafun.h*. These files are included in all the routines (see §IV.3). This allows more readability and more security: all the arrays appear at the same place, with a fixed amount of memory used; the statement functions are always properly placed (i.e. just before the first FORTRAN instruction). For clarity and modularity, the various computations done by the code are ordered as the terms of the physical equations to be solved. As a consequence, when modifying a physical process or of a discretization of one specific term, only one small routine is often to be changed (see §IV.1-b)

- **portability/performance**: No computer specific coding has been introduced except for parallelization (see §IV.1-c) and Input/Output (thereafter I/O) (see §III.12-a). The code is intended to run on any computer having enough memory and CPU power, and words of 64 bits or more. It is presently used for several benchmarks on super-computers. It is optimized for vectorial processing, weak and massive parallel processing while keeping its readability (i.e. readability always prevails over performances) (see §IV.2). Note that the minimum of 64 bits words is really required. A recent analysis of the code using CADNA software [Jézéquel and Chesneaux, **1998**] have shown that the lost of significant digits during a time-step computation is large enough to loose the physical meaning when using 32 bits words.

IV.1-b Flow Chart

The goal of the OPA numerical model is to compute a solution for the equations described in section I in a chosen space and time configuration. For that purpose, the code contains two parts. The first one is the initialisation phase. It is run once at the beginning of a numerical experiment, and ensures:

- physical and numerical parameter initialization.
- space and time domain definition: mesh, bathymetry, land points and time step setting.
- initialization for the surface pressure gradient computation, including islands.
- surface forcing initialization: wind stress, heat and fresh water fluxes.
- external temperature and salinity data input if necessary.
- definition of mixing coefficients, Newtonian damping coefficients, if necessary.
- dynamics and tracers initialization.

The second part is the time stepping. It starts at end of the initialization phase, as everything is ready to start for the time loop. It ensures :

- update of surface forcing and data
- update of ocean physics
- tracer and dynamics trends computation
- time stepping of prognostic variable (u, v, T, S)
- diagnostic variable computation ($\rho, N^2, \chi, \zeta, w$)
- diagnostics and/or output

model - main program

opa - main routine

First part : model settings

inipar - **ini**tialisation : model and file **par**ameters
- **parcst** - **par**ameters: **c**on**st**ants
- **parlec** - **par**ameters: **lec**ture of namelist
- **parctl** - **par**ameters: **c**on**t**ro**l** and monitoring of model options
  - **ctlopn** - **c**on**t**ro**l**: **op**e**n** file and check

inimpp - **ini**tialisation: subdomains **m**assively **p**arallel **p**rocessors

inidom - **ini**tialisation: **dom**ain
- **dommba** - **dom**ain: read the integer bathymetry array (**mba**thy)
- **dommsk** - **dom**ain: **m**a**sk**s
- **domhgr** - **dom**ain: **h**orizontal **gr**id
  - **hgrcoo** - read the **h**orizontal **gr**id **coo**rdinate file
- **domzgr** - **dom**ain: vertical (**z**) **gr**id
- **domstp** - **dom**ain: time-**st**e**p**s initialization

inispg - **ini**tialisation: **s**urface **p**ressure **g**radient
- **bsfmat** - **b**arotropic **s**tream **f**unction: **mat**rix
- **isldom** - **isl**and: **dom**ain check
  - **islpri** - **isl**and: **pri**nt
  - **islpth** - **isl**and: **p**a**th** around
- **islbsf** - **isl**and: **b**arotropic **s**tream **f**unction
  - **bsfpcg** - **b**arotropic **s**tream **f**unction **p**re**c**onjugate **g**radient solver
- **islmat** - **isl**and: **mat**rix

inidta - **ini**tialisation: **d**a**ta** fields (tracers and/or velocity)
- **dtatem** - **d**a**ta**: lecture of **tem**perature fields
- **dtasal** - **d**a**ta**: lecture of **sal**inity fields
- **dtacof** - **d**a**ta**: restoring **coef**ficients

inidtr - **ini**tialisation: **d**ynamics and **tr**acers
- **dtrlec** - **d**ynamics and **tr**acers: **lec**ture and initialization
- **dtrtem** - **d**ynamics and **tr**acers: **tem**perature initialization
- **dtrsal** - **d**ynamics and **tr**acers: **sal**inity initialization
- **eos** - **e**quation **o**f **s**tate: in-situ density
- **bn2** - **B**runt-Vaïsälä frequency N^2
- **prh** - **hy**drostatic **p**ressure
- **wzv** - **w**: vertical (**z**) **v**elocity

iniice - **ini**tialisation: **ice** model

initrc - **ini**tialisation: passive **tr**a**c**ers model

inihdf - **ini**tialisation: **h**orizontal **d**iffusive coefficients

inizdf - **ini**tialisation: vertical (**z**) **d**iffusive coefficients

inicmo - **ini**tialisation: **c**oupled **mo**dels

mlbxinit - task initialisation (macro-tasking option)

tskstart - call of step (macro-tasking option)

Second part : time stepping

step - time **step**ping loop

Update forcing and data
- **day** - model calandar (**day**)
- **tau** - wind stress (**tau**)
- **flx** - surface **flux**es (heat and fresh water fluxes)
- **dtasst** - **d**a**ta**: lecture of **s**ea **s**urface **t**emperature fields
- **dtatem** - **d**a**ta**: lecture of **tem**perature fields
- **dtasal** - **d**a**ta**: lecture of **sal**inity fields
- **diawri** - **dia**gnostic: **wri**te the standard output of opa at the first time step

Table IV.1 : model flow trace

Update ocean physics
- **zdftke** - vertical (**z**) **d**i**f**fusion: **tke** (1.5 turbulent closur scheme)
- **zdfric** - vertical (**z**) **d**i**f**fusion: **Ric**ihardson number dependent coef.
- **zdfevd** - vertical (**z**) **d**i**f**fusion: convection, **e**nhanced **v**ertical **d**iffusion
- **zdfbfr** - vertical (**z**) **d**i**f**fusion: **b**ottom **fr**iction
- **hdfslp** - **h**orizontal **d**i**f**fusion: **slop**e of the direction of lateral diffusion

Active tracer trends
- **trahad** - **tra**cer: **h**orizontal **ad**vection
- **trahdf** - **tra**cer: **h**orizontal **d**i**f**fusion
- **tradmp** - **tra**cer: newtonian **d**a**mp**ing
- **trazad** - **tra**cer: vertical (**z**) **ad**vection
- **traqsr** - **tra**cer: penetrative solar radiation **qsr**
- **trazdf** - **tra**cer: vertical **d**i**f**fusion

Sea-Ice and Passive tracer models
- **icestp** - **ice**: first time **st**e**p**ping routine (**1**)
- **iptrstp** - **p**assive **tr**acer model: time **st**e**p**ping

Dynamics trends
- **dynkeg** - **dyn**amics: horizontal **g**radient of **k**inetic **e**nergy trend
- **dynvor** - **dyn**amics: **vor**ticity term (including Coriolis term)
- **dynhdf** - **dyn**amics: **h**orizontal **d**i**f**fusion trend
- **dynhpg** - **dyn**amics: **h**orizontal **p**ressure **g**radient trend
- **dynzad** - **dyn**amics: vertical (**z**) **ad**vection trend
- **dynzdf** - **dyn**amics: vertical (**z**) **d**i**f**fusion trend
- **dynspg** - **dyn**amics: **s**urface **p**ressure **g**radient
  - **bsfpcg** - **b**arotropic **s**tream **f**unction **p**re**c**onjugate **g**radient solver
  - **bsfsor** - **b**arotropic **s**tream **f**unction **s**u. **o**ver **r**elax.

Active tracer and dynamics time stepping
- **tranxt** - **tra**cer: tracer fields at **nxt** time step
- **tranpc** - **tra**cer: **n**on-**p**enetrative **c**onvective mixing
  - **eos** - **e**quation **o**f **s**tate: in-situ density
  - **bn2** - **B**runt-Vaïsälä frequency N^2
- **dynnxt** - **dyn**amics: horizontal velocity fields at **nxt** time step

Diagnostic variables tracer and dynamics time stepping
- **eos** - **e**quation **o**f **s**tate: in-situ density
- **bn2** - **B**rünt-Vaïsala frequency N^2
- **div** - **div**ergence of the horizontal velocity
- **cur** - relative vorticity (**cur**l)
- **wzv** - **w**: vertical (**z**) **v**elocity

Outputs and diagnostics
- **stpctl** - **st**e**p**: **c**on**t**ro**l** of the simulation
- **diawri** - **dia**gnostic: **wri**te the standard output of opa
- **rstwri** - **r**e**st**art: **wri**te the standard restart file of opa
- **icedia** - **ice**: **dia**gnostic and outputs
- **ptrdia** - **p**assive **tr**acer model: **dia**gnostic and outputs
- **stpcpl** - **c**ou**pl**er: send SST and sea-ice extend

end
- tskwait - task get back (multitasking option)
- mppstop - stop and close (mpp option)

Table IV.1 : *continued*

The time loop stops at the last time step (specified in the *namelist* file) and the model stops after the creation of a restart file.

The code is organized in small routines which are devoted to one task only. For example, in the time loop integration, each subroutine computes only one trend term of dynamics or tracer equations. Each routine name is formed by three or six letters except for main programs (*model.F* and *step.F*) or computer dependant routines (macro-tasking or MPP routines, matrix computations, more than 6). These names are formed of two mnemonic root names of three letters. Main keys are :

- **ini** **ini**tialization
- **dom** **dom**ain
- **dyn, tra** **dyn**amics, **tra**cer
- **tem, sal** **tem**perature, **sal**inity
- **dta** **d**a**ta**
- **had, zad** **h**orizontal, vertical (**z**) **ad**vection
- **hdf, zdf** **h**orizontal, vertical (**z**) **dif**fusion
- **spg, hpg** **s**urface, **h**ydrostatic **p**ressure **g**radient
- **sol** elliptic **sol**ver
- **bsf** **b**arotropic **s**tream**f**unction
- **isl** **isl**and
- **ctl** con**t**ro**l**
- ...

The model flow chart is given in table IV.1. Some routines are not listed in this flow chart. They concern:
- array printings in the output listing: *prihre.F*, *prihin.F*, *prizre.F* (**pri**nt an horizontal or vertical (**z**) **re**al or **in**teger array), *islpri.F* (**pri**nt **isl**and information);
- I/O routines: *wrivr2.F*, *wrivr3.F* (**wri**te **V**ai**r**mer **2**D and **3**D file, respectively), *read2.F*, *read3.F*, *write2.F*, *write3.F*.
- macro-tasking synchronisation : *forkjoin.F*.
- the setting of model domain lateral boundary conditions : *lbc.F*.
- exchange or sum of data when using the massively parallel option : *mppsend.F*, *mpprecv.F* (**send** and **rec**ei**v**e data to neighbouring processors in **m**assively **p**arallel **p**rocessors computation), and also *mppsum.F*, *mppgather.F*, *mppscatter.F*, *mppsync.F*, *mynode.F*.

Although these routines are not important for the general understanding of the code, they are *very* important if a user wants to modify the code structure since they synchronize the computation when using more than one processor (i.e. when **key_monotasking** is not defined or **key_mpp** is defined, see §IV.2).

IV.2 PERFORMANCE — PORTABILITY

Due to the large number of horizontal grid points and the increasing time of integration, the limitation of computer time is a severe restriction on numerical ocean simulations. Therefore numerical experiments must be performed on the fastest available machines. Today two types of powerful computers coexist: (1) supercomputers (one or a few vectorial processors which access to a shared memory), (2) massively parallel computers (some ten or hundred scalar or vectorial processors, each processor having its own local memory). The present release of OPA can be run on these two types of computers, and is optimized for vectorial processors.

IV.2-a Vectorization

Supercomputers achieve great performances thanks to the architecture of their processors. Their fast speed is the result of their processors being able to compute on long chains of variables (vectors), as on an assembly line. Some care is taken in the code programming to ensure that the machine will be able to recognize long vectors (some tens to hundreds of numbers) to speed-up the calculation. This is called vectorization. The vectorization is implemented in a classical way, respecting the general requirement for vector computers. For instance, the use of IF-testing within DO loops is avoided, since it often prevents fast arithmetic processing, and the inner loop index is (as far as possible) the leftmost argument in order to use the parallel reading of the core memory.

On Cray computers, several routines are available to optimize vector computation of specific operations. Only a few of them are used in OPA: *sdot* (dot product), *cvmgp* (comparison of two vector elements to construct a third one), *folr* (first order recurrence, see §III.5), ... Their FORTRAN equivalent code is available in LODMODEL/SRC_AUX directory (see §IV.3) and can be used to run the code on non-Cray computers.

The vectorization speed up rate of OPA (i.e. the ratio between vector and no vector computation CPU time) is between 10 and 17, depending on the size of the model domain (larger values of *jpi* usually produce greater speed up). Note that since the code is optimised for vector processors, it is not fully optimized for scalar ones (such as those used today on Cray T3D and T3E massively parallel computers).

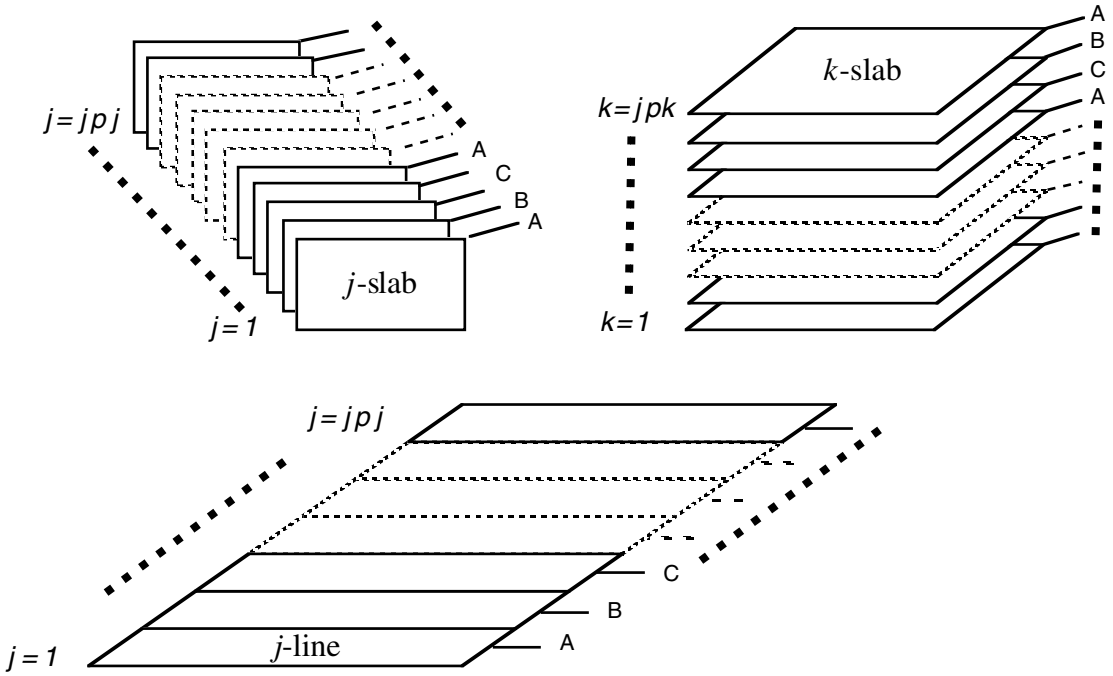


Figure IV.1 distribution of *j*- and *k*-slabs on three processors A, B, and C

IV.2-b Weak Parallelism

(default option , inhibited if **key_mpp** or **key_monotasking** is defined)

Supercomputers generally have a few CPUs which access to a shared memory. The use in parallel (i.e. at the same time) of a few CPUs to speed up a simulation by decreasing the overall elapse time is called multitasking (or weak parallelism). The FORTRAN compiler now available on such computers can generaly recognize the intrinsic parallelism of a FORTRAN code (autotasking option of the compiler). It distributes each do-loop instruction on the free processors. Nevertheless, we have chosen to maintain the algorithmic and programming efforts made by Andrich *et al.* [**1988*a***, **1988*b***] to take advantage of a weakly parallel computers. It basically consists in distributing large portions of the computer code on each processor (macrotasking). Such multitasking computation is active by default, and removed if either **key_monotasking** or **key_mpp** cpp variable is defined. Note that weak parallelism slightly increases the total CPU time of an experiment. The benefit of its use thus depends on the computer center policy and on the attention devoted to the elapse time. Usually, the macrotasking option is advised when the model configuration requires half or more of the shared memory.

The computer code has been macrotasked in an operator oriented way: the macrotasking is implemented in a different way for each of the integro-differential operators of the equations, using a data partition technique. Each term of the model equation is distributed on the available processor in a way that depends on its nature. Usually, horizontal trends are computed on *k*-slabs (i.e. (i, j) plane), vertical ones are computed on *j*-slabs (i.e. (i,k) plane) while two dimensional ones are computed on *j*-lines (Fig. IV.1). In all cases, the inner do loop remains the *i*-index which allows a better efficiency for vectorization. Synchronization points are generally required between each type of distribution.

The implementation is rather easy due to the structure of the computer code. Each term of the model equations is associated with a specific routine and called by one routine (*step.F*). Therefore, most of the synchronization points have to be settled in this routine (Table IV.2). Nevertheless, a few routines require internal synchronizations as they include horizontal and vertical operator, namely *zdftke.F*, *tranpc.F*, *dynspg.F* , and *trahdf.F* or *dynhdf.F* (when a rotated biharmonic lateral diffusive tensor is used). In particular, *dynspg.F* solves an elliptic equation either with a successive over-relaxation or a conjugate gradient method (§III.5). The former is only partly parallelized while the latter is fully parallelized following Oppe and Kincaid [1987].

Each synchronisation point used during the macrotasked computation is made with a call to the *forkjoin.F* routine: the first processor which enters this routine establishes a barrier which stops the other processors. When all the processors have entered this routine, the barrier is opened so that all the processors can resume the execution of the code. *forkjoin.F* contains a single instruction, a call to a computer specific routine which ensure the synhronization described above. In standard, the routine called is barsync, a Cray computer library routine. When another weakly parallel super-computer is used, only the call to barsync in *forkjoin.F* should be changed.

The code remains easy to understand as the macro-tasking is rather transparent to the user. In particular, the portions of the computer code that the user generally must modify for a specific application (i.e. the initialization phase, and the surface forcing, data updating, diagnostics and outputs at each time step) have been kept on one processor in order to simplify user's work, even if this leads to a slight lost of elapse time speed up.

forcing and data updating:	
stpday, **tau**, **flx**, **dtatem**, **dtasal**, **dtasst**	— ONE processor
ocean physics:	
zdftke	*k*- and *j*-slabs (with synchronization points)
zdfbfr	*j*-slabs
hdfslp	*k*-slabs
tracer trends:	
trahad, **tradmp**, **trahdf**	*k*-slabs
trazad, **traqsr**, **trazdf**	*j*-slabs
dynamics trends:	
dynkeg, **dynvor**, **dynhdf**	*k*-slabs
dynhpg, **dynzad**, **dynzdf**	*j*-slabs
dynspg	*j*-lines (with synchronization points)
time stepping	
tranxt	*k*-slabs
dynnxt	*k*-slabs
diagnostic variables:	
eos, **bn2**, **div**, **cur**	*k*-slabs
wzv	*j*-slabs
control, outputs, diagnostics:	
stpctl, **diawri**, **rstwri**	— ONE processor

Table IV.2 : Synchronization points and type of distribution over the processors used in macro-tasking option for the time stepping routine (step.F). Note that the example is for the most commonly use cpp options, i.e. default option + key_zdftke + key_flxqsr.

IV.2-c Massive Parallelism (**key_mpp** defined)

The traditional vector supercomputers seem to approach their technological limits in term of speed and memory. The Massively Parallel Processor (MPP) computers, characterized by a large number of processors and a communication network, are likely to be a way to reach teraflops speed and teraoctet central memory. For this reason, many meteorological or oceanological centres and laboratories have adapted their general ocean circulation models for MPP architecture [Bleck *et al.* 1995, Guyon **1995**, Webb 1996, Beare and Stevens 1997, Guyon *et al.* **1999**].

A first attempt for using OPA model on massively parallel machine was performed in 1993 on a CM2 Connection Machine, i.e. onto a SIMD machine (Single Instruction Multiple Data - all the processors execute the same computation at the same time and all the data stored in the global memory are available for each processor). This adaptation of the code has been realized through a collaboration of CEA, LMCE and LODYC [Vittart 1993], using a specific language, CMFortran, which introduces statements and functions for distributing or accessing the shared data. Many code lines were rewritten, especially the array dimension and do loop, in order to obtain a reasonable efficiency. All the code was adapted except the surface pressure gradient computation.

Since then, SIMD machines were substituted by MIMD machines (Multiple Instruction Multiple Data), another approach was chosen: a domain decomposition method. The basis of the method consists in splitting the large computation domain of a numerical experiment into several smaller domains and solving the set of equations by addressing independent local problems. Each processor has its own local memory and computes the model equation over a sub-domain of the whole model domain. The sub-domain boundary conditions are specified through communications between processors which are explicitly organized by specific statements (message passing method). Advantages are that there is not many modifications of the initial FORTRAN code. For the modeller's point of view, each sub-domain running on a processor is identical as the old mono-domain code. In addition, the programmer manages the communications between sub-domain, and the code presents more scalability when the number of processors is increased. The porting of OPA code on an iPSC860 was achieved during Guyon's PhD [Guyon *et al.* **1994a**, **1994b**] in collaboration with CETIIS and ONERA. The implementation in the operational context and the studies of performances on a T3D and T3E Cray computers have been made in collaboration between IDRIS and LODYC. The present implementation is largely inspired from Guyon's work [Guyon 1995, Guyon *et al.* **1999**].

As for the macro-tasking technique, the parallelization strategy is defined by the physical characteristics of the ocean model. Second order finite difference schemes leads to local discrete operators that depend at the very most on one neighbouring point. The only non-local computations concerne the vertical physics (i.e. implicit diffusion, *1.5* turbulent closure scheme, ...) which involves the whole water column, and the solving of the elliptic equation associated with the surface pressure gradient computation which involves the whole horizontal domain. Therefore, a pencil strategy is used for the data sub-structuration: the 3D initial domain is laid out on local processor memories following a 2D horizontal topological splitting. Each sub-domain computes its own surface and bottom boundary conditions. It has a side wall overlapping interface which stocks lateral boundary conditions for computations in the inner sub-domain. The overlapping area is reduced to one row. After a computation, a communication phase starts: each processor sends to its neighbouring processors the update values of the point corresponding to the overlapping area of its neighbouring sub-domain. The communication is done through message passing. Usually the parallel virtual language, PVM, is used as it

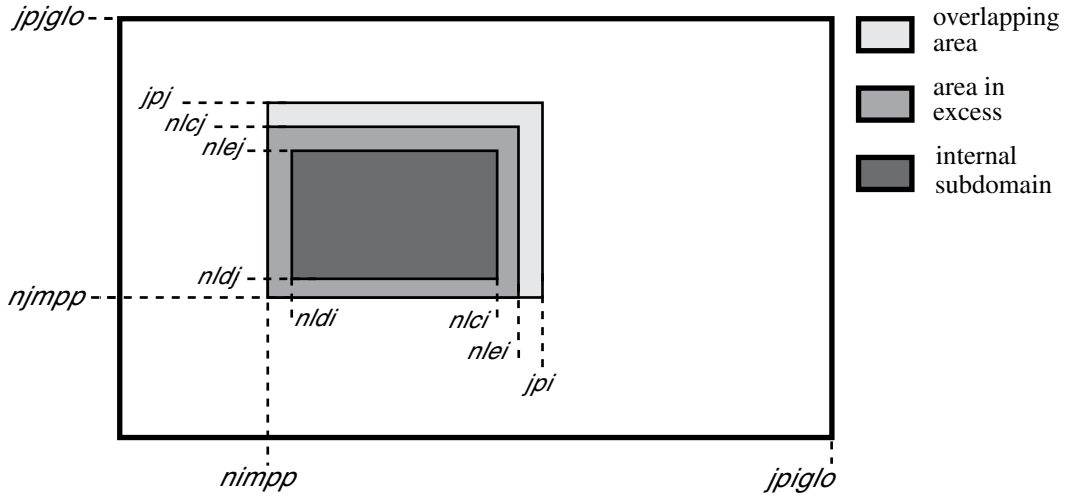


Figure IV.2: positioning of a sub-domain.

is a standard language available on nearly all MPP computers. More specific languages (i.e. computer dependant ones) can be easily used to speed up the communication, such as SHMEM on the Cray T3E computer. The data exchanges between processors are required at the very place where lateral domain boundary conditions are set in the mono-domain computation (§III.10-c): the *lbc.F* routine which manages such conditions is substituted by *mpplnk.F* or *mpplnk2.F* routine when running on MPP computer (**key_mpp** defined). It has to be noticed that when using MPP version of the model, the east-west cyclic boundary condition is implicitly done, while the south-symmetric boundary condition option is not available.

In the standard version of the OPA model, the splitting is regular and arithmetic. the *i*-axis is divided by *jpni* and the *j*-axis by *jpnj* for a number of $jpni \times jpnj$ processors (model parameter set in *parameter.h*). Each processor is independent. Thus, without message passing or synchronous process, programs run alone and only access to its own local memory. For this reason, the main model dimensions are now the local dimensions of the subdomain (pencil) that are noted *jpi*, *jpj*, *jpk*. These dimensions include the internal domain and the overlapping rows. The number of overlapping rows is usually set to one (*jpreci=1*, in *parameter.h*). The whole domain dimensions are named *jpiglo*, *jpjglo* and *jpk* (see §III.12). The relationship between the whole domain and a sub-domain is for the abscisses :

$$jpi = (jpiglo - 2 \times jpreci + jpni - 1) / jpni + 2 \times jpreci$$

where *jpni* is the number of processors along the *i*-axis.

One defines also variables *nldi* and *nlei* which correspond to the internal domain bounds, and the variable *nimpp*, *njmpp*, the global position of the (*1*,*1*) gridpoint, *i.e.* an element of T_l, a local array (sub-domain), corresponds to an element of T_g, a global array (whole domain), by the relationship:

$$T_g(i + nimpp - 1,\ j + nlmpp - 1,\ k) = T_l(i,\ j,\ k)$$

with $1 \le i \le jpi$, $1 \le j \le jpj$, and $1 \le k \le jpk$.

The processors are numbered from *0* to $jpni \times jpnj - 1$. This number is saved in the variable *nproc*. In the standard version, a processor has no more than four neighbouring processors named *nono* (for north), *noea* (east), *noso* (south) and *nowe* (west) and two variables, *nbondi* and *nbondj*, indicate the situation of the processor (see Fig.IV.3). For example, *nbondi* can take the following values depending on the existence of east and/or west processors:

nbondi = *-1* an east neighbour, no west processor,
nbondi = *0* an east neighbour, a west neighbour,
nbondi = *1* no east processor, a west neighbour,
nbondi = *2* no splitting following the *i*-axis.

During a simulation, the processors exchange data with their neighbours. If there is effectively a neighbour, a processor receives variables from this processor on its overlapping row, and sends the data issued from internal domain corresponding to the overlapping row of it (Fig. IV.4).

As lands usually occupy a non-negligible portion of a model domain, it often happens that at least one sub-domain contains only land grid points. In the standard case, the computation is still done on this processor, but with the use of the mask technique (see III.1-c) all the

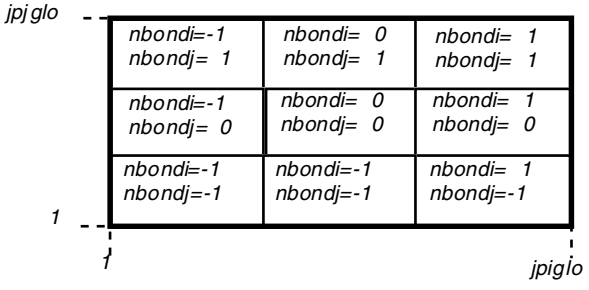


Figure IV.3: example of a domain splitting with 9 processors and no east-west cyclic boundary conditions.

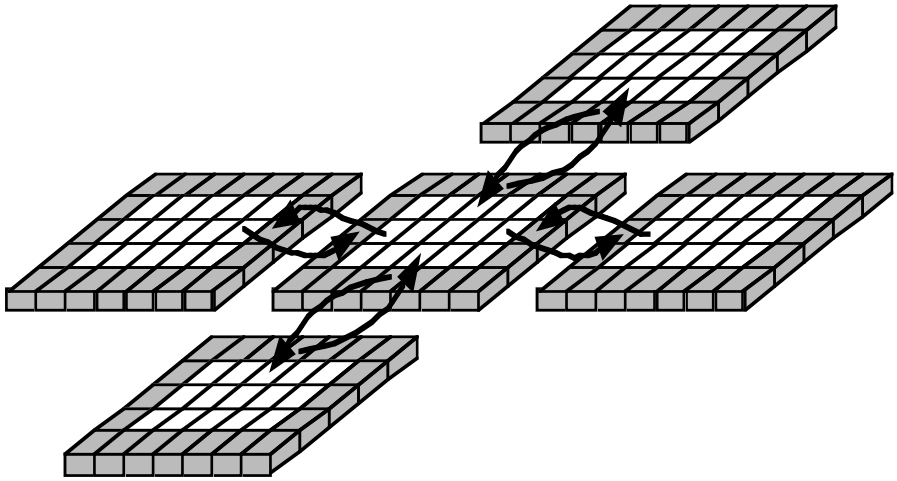


Figure IV.4: Pencil splitting with the additional outer halos.

variables remain zero on the associated sub-domain. An alternative is to not consider the land-only processors, and thus only use a number of processors equal to the number of sub-domains that contain ocean grid-points. For that purpose, a pre-processing tool has been developped. This tool (*mpp_optimiz.f* which can be found in LODMODEL/SRC_AUX/AUX_OPA directory) works as follows:

The user provides a maximum number of processors to use, the size of the model domain (*jpiglo*, *jpjglo*, *jpk*), the *bathymetry* file which defines the model domain, and the size of the in core memory of the mpp computer processors (let say for example, 180 processors, *jpiglo*=773, *jpjglo*=1236, *jpk*=42, for the 1/6° Atlantic domain (Fig. IV.5) on a T3D for which each processor has 16 Mw of in core memory). The tool search all the

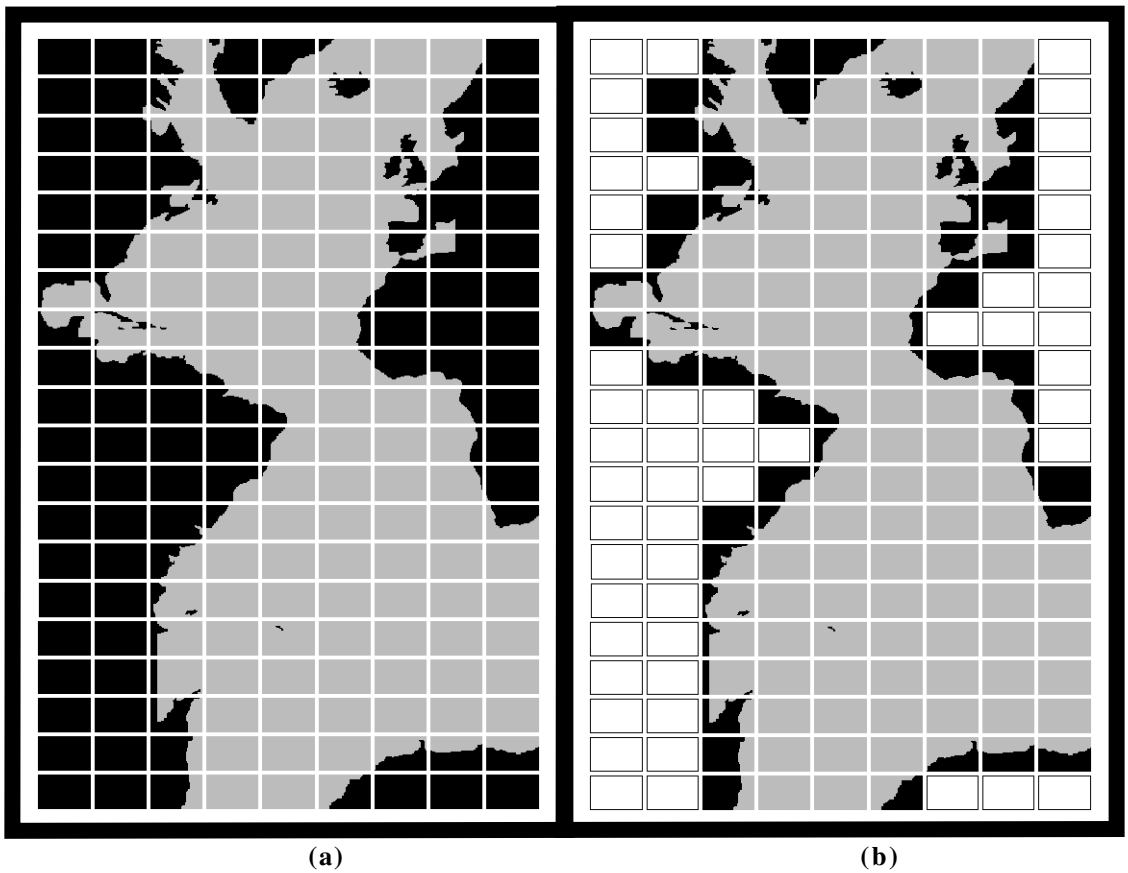


Figure IV.5: Example of the Atlantic domain defined for the CLIPPER experiments. The iitial grid is composed of 773 x 1236 horizontal points. (a) the domain is split onto 9 x 20 subdomains (jpni=9, jpnj=20). 52 subdomains are land areas. (b) 52 subdomains are eliminated (white rectangles) and the resulting number of processors really used during the computation is jpnij=128.

domain decompositions that respect the in core memory size of the local processors and class them by decreasing number of land only processors. The user chooses one decomposition, let say a splitting in 9 by 20 in the *i*- and *j*-direction which has 52 land-only sub-domains (Fig. IV.5). Then, he indicates in the *parameter.h* file the chosen cutting along *i*- and *j*-axes (*jpni* and *jpnj* values) and the number of ocean processors (*jpnij* < *jpni* x*jpnj*). Each processor name and neighbour parameters (*nbound*, *nono*, *noea*,...) are modified by an algorithm in the *inimpp2.F* so that the model will be run on *jpnij* processor only, not on *jpni*x*jpnj* processors.

IV.2-d Code Requirements and Performances

The OPA model needs to be run on a computer with at least 64-bit words in order to get the necessary numerical precision. Otherwise the truncature error becomes unacceptably large (especially in the implicit resolution of vertical diffusion terms and pressure gradient terms).

In terms of memory, given the size of the grid (i.e. the number of longitude points *jpi*, the number of latitude points *jpj* and the number of depth levels *jpk*), a rough estimation of the requested memory will be :

Memory needed = $55(jpi \times jpj \times jpk) + 73(jpi \times jpj)$ words

or 8 times more if you want the result in bytes). This memory request will change as soon as you add or suppress arrays in the code.

In terms of CPU, it will strongly depend on the amount and efficiency of the diagnostics used, but a rough estimation would be: for a rectangular oceanic box with very few diagnostics : $1.82\ 10^{-6}$ seconds per time-step and grid point on a CRAY C90 ; for a real experiment with some diagnostics: $2.13\ 10^{-6}$ seconds per time-step and grid point on a CRAY C90, which corresponds to a *390* Mflops performance.

Since the OPA model has been since a long time used for benchmarks in scientific computing, we have some results of the code performances on various computers. For example, one can look at the Banc d'essai MIPS, a paper published (in french!) in AFUU Dossier spécial Benchmarks n°4 Mars 93.

More recently, the OPA code has been tested on massively parallel systems, and its performance compared to the C90's as described below, using the work of M. Imbard (LODYC), J. Escobar (IDRIS) and M. Guyon (CETIIS):

Performances for a simple rectangular grid (*jpi= 164* , *jpj=164*, *jpk=31*)

1) Comparing performances of the code kernel (no I/O) on one vector processor of CRAY C90 and FUJITSU VPP700 .

	Time (seconds)	Mflops	Ratio
C90	226.41	497	
VPP	212.00	531	1.07

2) Comparing performances of the code kernel (no I/O) using PVM on different multiprocessors computers

For 16 PEs, the local size of the model on one processor is 44x44x31 grid-points, which results in a quite poor vectorization (the maximum vector length is 44).In addition, the PVM version on VPP is two times faster then on the T3D, which becomes as good as the SHMEM version.

COMPUTER	elapsed time (seconds)	Communi-cation time (seconds)	Mflops
VPP700 16 PEs PVM	136	20	1813
T3D 64 PEs SHM	603	11	409
C90 1 CPU	609	0	405
T3D 64 PEs PVM	636	44	387
T3D 16 PEs SHM	1885	17	131
T3D 16 PEs PVM	1876	51	131

3) Scalability tests on T3E with SHMEM message passing

number of processor	32	64	128	256
Time (seconds)	386	198	98	50
Speed-Up	1.0	1.95	3.94	7.65
Efficiency	100%	98%	98%	96%

References

(see OPA-Bibliography when the year is in **bold**)

Beare M. I., and D. P. Stevens, 1997: Optimization of a parallel ocean general circulation model. Ann. Geophysicae, 15, 1369-1377.

Bleck, R., M. O'Keefe, and A. Sawdey, 1995: A comparison of data-parallel and message-passing versions of the Miami Isopycnic Coordinate Ocean Model (MICOM). Parallel Computing, Special issues on applications, climate and weather modeling, 21, 10.

Oppe T.C., and D.R. Kincaid, 1987: Numerical Experiments with a Parallel Conjugate Gradient Method, *report from the Center for Numerical Analysis,* University of Texas.

Webb, D. J., 1996: An ocean model code for array processor computers. Computers & Geosciences, *22*, 5, 569-578.

IV.3 ENVIRONMENT

IV.3-a UNIX Environment of the Model

As mentioned in §IV.1, OPA computer code has been developed with a well thought-out goal about portability: it can be performed on any computer which has a FORTRAN compiler. However, the code is used at LODYC in a UNIX environment and for the management, UNIX facilities are employed, namely the C Pre-Processor (cpp) and the make command. The cpp improves the legibility and optimizes the number of lines of the source code. It is used to:

(1) insert some parts in the source code (through #include) (Fig. IV.6a), especially for the parameter and common array declarations, and the statement function definition. All include files are suffixed by *.h*, so that they are easily identified, whereas routine with include orders are suffixed by *.F* (*.f* otherwise).

(2) allow sections of the FORTRAN code to be selectively turned on or off (through #define condition or #if defined condition and #endif) (Fig. IV.6b).

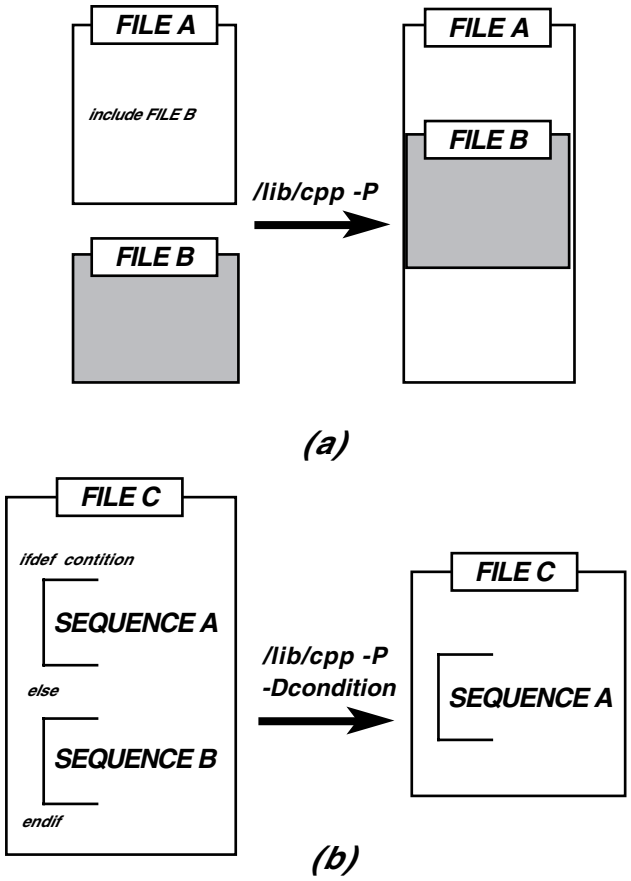


Figure IV.6 : C Pre-Processor (cpp) action for (a) #include command and (b) #defined command

The cpp is used before entering the compilation phase and is controlled by cpp variables (conditions) defining the chosen options. The cpp keys used in the model correspond to algorithms, formula and specific conditions chosen for a particular simulation (see §II and §III).

Files

- The computer code consists in some FORTRAN algorithms, FORTRAN programs or UNIX statements which are stored in files with different suffixes :

- **xxxx.f** FORTRAN source files (ex : prihre.f)
- **xxxx.F** main files before cpp pre-processor call (ex : opa.F)
- **xxxx.h**code or algorithm included in xxxx.F by cpp (ex : common.h)

The code contains almost 130 routines insepared files. They can be found in the LODMODEL/SRC_OPA directory (see LODMODEL/README_files for a complete description of file location).

IV.3-b How to Set Up the Model

In general, a user wants to configure the model for a specific ocean basin by choosing grid sizes and dimensions, defining its bathymetry and the forcing (i.e. the surface boundary conditions). This means that this user does not need to deal with the whole code, but just a very small part of it. The way to use the code, as we suggest it, intends to help the user by minimizing the amount of code he has to explicitly deal with.

The general idea is to start with a reference code located in a reference directory. This reference code is the same for all users and will NEVER be modified until a new release. Each user has its own directory containing ONLY the routines that he needs to change. During the compilation, these routines - usually just a few - will overwrite their counterparts of the reference code in the user's working directory on the computer.

In order to use the code, one must first check that it is suitable for the project. It is therefore strongly suggested to first discuss of the fit between the project and the OPA model with the scientists of the OPA development team. If this has not been done yet, please send an e-mail to delecluse@lodyc.jussieu.fr or madec@lodyc.jussieu.fr.

The second step is to get the tarfile in order to install the source files on your system. This tarfile contains the reference directory as described above. It has to be installed once and for all in a secured location where no third party can change it. This directory can be read-only since it should not be modified because the modified

routines and scripts will be in the user's directory. Once your choice has been made for this location, you can ask for the tarfile by sending an e-mail to levy@lodyc.jussieu.fr or imbard@lodyc.jussieu.fr.

The third step is to extract the reference directory from the tarfile and install it on your computer. The top directory is called LODMODEL. All further step should be documented in README files but the beginning will be:
- fill in and send back to the LODYC the OPA_agreement located in LODMODEL
- set up your own configuration (for that, start with the READMElocated in LODMODEL)
- the location and content of all files is described in the file README_files located in LODMODELdirectory
- please make sure that we can exchange information by subscribing to the opalist mailing list. In order to do this, please send an e-mail with no subject to listserv@lodyc.jussieu.fr containing the following line:

SUB opalist Your_Name

All further steps are self-documented (please tell us if you think some information is missing). But in order to help you find your way through all this, you will find below some specific details. The standard procedure, though, is to follow the information from the README files.

A given simulation of the model is led by several definitions or choices as constants, domain and grid, initial data and optional cpp keys:

constants

The constants are defined in the *parcst.F* file (corresponding to the FORTRAN program parcst) and are communicated to every subroutine by the common *comcst* included in the *common.h* file. The constants are defined as :

- mathematical constant

π : $rpi = 2 \arcsin(1.)$

- astronomical constants

day : $rday = 24 \times 60 \times 60$ **seconds**

sideral year : $rsiyea = \dfrac{365.25 \times rday \times 2 \times rpi}{6.283076}$ **seconds**

sideral day : $rsiday = \dfrac{rday}{(1 + rday / rsiyea)} 6.283076$ **seconds**

earth angular speed : $omega = \dfrac{2 \times rpi}{rsiday}$ **seconds**$^{-1}$

- geoid

earth radius : $ra = 6371229$ *meters*

gravity : $g = 9.80665$ **meters** $\times$ **second**$^{-2}$

- thermodynamical constants

fusion point : $rtt = 273.16$ °**Kelvin**

zero Celsius : $rt0 = 273.13$ °**Kelvin**

- conversion factors

degree into radian : $rad = rpi / 180.$

mm/day into m/s : $rcs = rday \times 10^{-3}$

domain and grid

The spatial dimensions *jpiglo*, *jpjglo* (horizontal grid dimension), and *jpk* (number of level) as well as the number of islands for a model configuration must be supplied in the *parameter.h* file.

The horizontal grid coordinates are defined or read in the *domhgr.F* file (see §III.2)

gphi and *glam*, 2D arrays allow to use a stretched orthogonal curvilinear grid.

The vertical levels are defined in the *domzgr.F* file (see §III.2).

The land points and bathymetry are computed or read in the *dommba.F* file (see §III.2).

cpp keys

Physical and algorithmic options that require a modification in the common are defined through cpp keys: these options are chosen before compilation of the FORTRAN source code and are described on line in LODMODEL/DOC/README.cpp file (see also the index at the end of the reference manual). All the cpp key name start by key_ in order to avoid problems with the use of makefiles.

Consistency between the chosen options for a simulation is checked in the *parctl.F* routine.

namelist file

Characteristics of a run (the number of time steps, the value of the horizontal diffusion coefficient ...) are defined through namelist statements (see the index). They are described in the LODMODEL/SRC_OPA/namelist file. They are taken in account at the execution of the model, thus changing one of those values does not imply a new compilation of the code (on the opposite, a change in cpp option modifies the FORTRAN sources and thus require a new compilation).

filenames

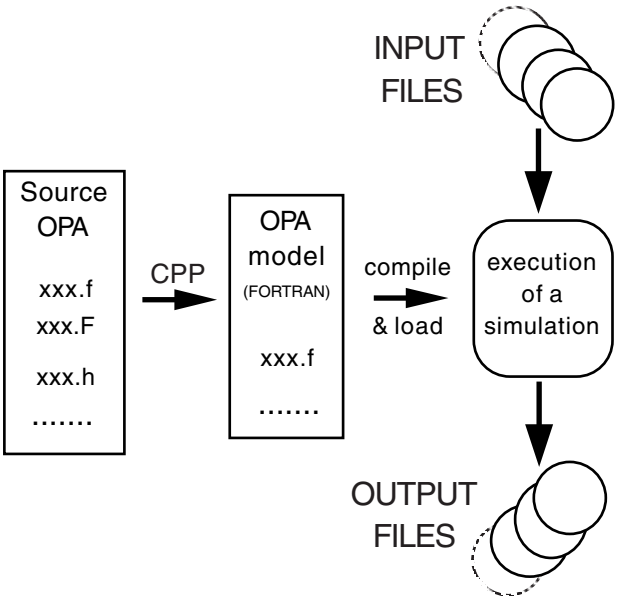
The input and output data sets are explicitly defined in the *parctl.F* routine. Therefore, the user has no choice in the naming of the files used in the standard version: each file has a standard name defined in the OPEN FORTRAN statement associated to a logical unit. If you need to add new input or output files, the complete list of logical units and associated standard names can be found in the header of the *parctl.F* routine. The used logical units are printed in the standard output of each run.

IV.3-c How to Run the Model

To summarize, when a user wants to set up its own model, he or she has to :

- define its grid size (in *parameter.h*)
- define its grid function (in *domhgr.F* and *domzgr.F*)
- define the land points and bathymetry (in *domlec.F*)
- select the set of cpp variables (while running the fait_make_remote command, see LODMODEL/README)
- check the values of namelist variables (namelist file)

In order to run a numerical simulation, your own OPA fileset must be conditioned by the cpp key you have chosen. It can then be compiled, loaded and executed with all your input files and the numerical values you have chosen in the namelist file :



To generate the correct makefile and run the code, please follow carefully the README located in the LODMODEL directory.

Some test examples can be found in LODMODEL/DOC/EXAMPLES. Each example is presented in a separate directory (see the README file), with the modified OPA routines, the output listing, and the graphic outputs build using the LODMODEL/UTILS/OPADRA graphic package.

IV.3-d How to Modify the Model

Since every authorized user has access to the whole source code, he or she may modify or add source files anytime. In order to be able to take into account your own needs and developments for further releases, we ask you to follow our coding rules:

- concerning the FORTRAN variables naming conventions, please see appendix D
- if you develop new parts of the code, please do not create new cpp keys: it would not be secure if we have to include developments from other users at the same time because we might get to conflicting codes. So please write your code with standard FORTRAN IF, and we will add a new cpp key when including your developments in the reference code if necessary
- if you need to declare variables or arrays, please use the common.h file rather than passing them through subroutine arguments
- if you want your code to be tested by other users, do not hesitate to annonce your developments through the mailing list opalist@lodyc.jussieu.fr

OPA-BIBLIOGRAPHY

PAPERS

1988

Andrich P., G. Madec, and D. L'hostis, 1988: Performance evaluation for an Ocean General Circulation Model, Vectorization and Multitasking. In *Conference Proceedings of the 1988 International Conference on Supercomputing*, July 4-8, St-Malo, France, AMC press, 295-303.

Andrich P., P. Delecluse, C. Lévy, and G. Madec, 1988: A multitasked general circulation model of the ocean. In *Science and engineering on Cray Supercomputers*, Proceedings of the 4th International Symposium Minneapolis, Minnesota, Cray Research, Inc. book, 407-428.

Madec G., C. Rahier and M. Chartier, 1988: A comparison of two-dimensional elliptic solvers for the barotropic streamfunction in a multilevel OGCM. *Ocean Modelling*, 78.

Merle J., and A. Morlière, 1988: Toward a 3 dimensional simulation of the tropical Atlantic. *Geophys. Res. Lett., 15*, 653-656.

1989

Lott F., and G. Madec 1989: Implementation of bottom topography in the Ocean General Circulation Model OPA of the LODYC: formalism and experiments. *Int. Rep.*, LODYC, No 3, 36pp.

Morlière A., G. Reverdin and J. Merle, 1989: Assimilation of temperature profiles in a general circulation model of the tropical Atlantic. *J. Phys. Oceanogr, 19*, 1892-1899.

Morlière A., P. Delecluse, P. Andrich and B. Camusat, 1989: Évaluations des champs thermiques simulés par un modèle de circulation générale océanique dans l'Atlantique tropical. *Oceanol. Acta, 12*, 1, 9-22.

1990

Delecluse, P., 1990: Modélisation océanique pour les études de climat. In *Proceedings of the space & sea colloquium*, Paris, 24-26 sept. ESA SP-312, 209-214.

Lott F., G. Madec, and J. Verron, 1990: Topographic experiments in an Ocean General Circulation Model. *Ocean Modelling*, 88.

Madec, G., and O. Marti, 1990: Traitement des îles dans le modèle de circulation générale OPA. *Tech. Rep.* 90/03, LODYC, Paris, France, 29pp.

1991

Madec G., and M. Crépon, 1991: Thermohaline-driven deep water formation in the Northwestern Mediterranean Sea. In *Deep convection and deep water formation in the oceans*, P.C. Chu and J.C. Gascard (Eds.), Elsevier Oceanographic Series, 241-265.

Madec G., M. Chartier and M. Crépon, 1991: Effect of thermohaline forcing variability on deep water formation in the Northwestern Mediterranean Sea: a high resolution three-dimensional study. *Dyn. Atmos. Oceans, 15*, 301-332.

Madec G., M. Chartier, P. Delecluse and M. Crépon, 1991: A three-dimensional numerical study of deep water formation in the Northwestern Mediterranean Sea. *J. Phys. Oceanogr, 21*, 1349-1371.

Reverdin G., P. Delecluse, C. Lévy, A. Morlière, and J. M. Verstraete, 1991: The near surface Atlantic in 1982-1984: results from a numerical simulation and a data analysis. *Prog. Oceanog.*, *27*, 273-340.

1992

Arnault, S., A. Morlière, J. Merle, and Y. Ménard, 1992: Low-frequency variability of the tropical Atlantic surface topography: altimetry and model comparison. *J. Geophys. Res.* ,*97*, 14,259-14,288.

Delecluse, P., 1992: Modeling of oceans circulation. In *Energy and Water Cycles in the Climate System* , NATO Advanced Study Institute, Edited by E. Raschke and D. Jacob, Luneburg, Germany, 30 sept-11 Oct. 91, 235-260.

Delecluse P., 1992: Ocean and climate. *The Courier, 133, May-June 92, ACP-EEC, 28-30.*

Delecluse P., 1992: Equatorial ocean dynamics. *École d'été de l'INSU "Dynamique des fluides géophysiques", Roscoff, août 91, 191-199.*

Marti O., G. Madec and P. Delecluse, 1992: Comment on "net diffusivity in ocean general circulation models with nonuniform grids" by F. L. Yin and I. Y. Fung. *J. Geophys. Res.*, *97*, 12,763-12,766.

1993

Blanke B., and P. Delecluse, 1993: Variability of the tropical Atlantic ocean simulated by a general circulation model with two different mixed layer physics. *J. Phys. Oceanogr., 23*, 1363-1388.

Caniaux, G., H. Roquet, and S. Planton, 1993: A 3D mesoscale simulation of the ocean using data from the ATHENA88 field experiment. *J. Mar. Syst. 4*, 197--216.

Delecluse P., 1993: Circulation générale océanique. *École d'été CNES " Les climats subtropicaux et leur évolution: de l'observation spatiale à la modélisation", La-Londe-Les-Maures, 14-25 Sept. 92., 121-140.*

Delecluse P., 1993: Modelling the ocean circulation. in Long Term Climatic Variations - Data and Modelling, NATO Advanced Study Institute, Edited by J.C Duplessy, Sienne-Italy, Oct. 92.

Delecluse P., 1993: Variabilité basse fréquence du système couplé océan-atmosphère. École d'été CNES " Les climats subtropicaux et leur évolution: de l'observation spatiale à la modélisation", La-Londe-Les-Maures, 14-25 sept 92, 81-97.

Stockdale T., D. Anderson, M. Davey, P. Delecluse, A. Kattenberg, Y. Kitamura, M. Latif and T. Yamagata, 1993: Intercomparison of tropical Pacific ocean GCMs. WCRP 79, WMO /TD-545, 43pp plus figs.

1994

Angot P., and M. Laugier, 1994: La méthode FIC de raccordement conservatif de sous-domaines emboîtés pour un modèle de circulation océanique. C. R. Acad. Sci. Paris, t. 319, série II, 993-1000.

Brossier, F., L. Monier, and P. Delecluse, 1994: Simulation of long equatorial waves in the Pacific Ocean in relation with sea level oscillations and zonal mean currents. Oceanol. Acta, *17*, 4, 461-478.

Delecluse, P., J. Servain, C. Lévy, K. Arpe, and L. Bengtsson, 1994: On the connection between the 1984 Atlantic warm event and the 1982-1983 ENSO. Tellus, *46A*, 448-464.

Guyon, M., F.-X. Roux, M. Chartier, and P. Fraunié, 1994: A domain decomposition method applied to the baroclinic part of an OGCM on a MIMD machine. Ocean Modelling, 103.

Guyon, M., F.-X. Roux, M. Chartier, and P. Fraunié, 1994: A domain decomposition solver to compute the barotropic component of an OGCM in the parallel processing field. Ocean Modelling, 105.

Laugier, M. L. Mortier, and I. Dekeyser, 1994: Un modèle bidomaine aux équations primitives en océanographie physique. Oceanol. Acta, 17, N4, 355-367.

Servain, J., A. Morlière, and C. S. Pereira, 1994: Simulated versus observed sea surface temperature in the tropical Atlantic Ocean. The global atmosphere and ocean system, *2*, 1-20.

1995

Delecluse P., Boulanger J.P., Maes C. and C. Lévy, 1995: Modelling and observing the equatorial ocean dynamics. Operational Oceanography and satellite Observations", Ed. Gerard, CNES/SFM, Biarritz, 16-20 Oct. 95.

EUROMODEL Group (P.M. Lehucher, L. Beautier, M. Chartier, F. Martel, L. Mortier, P. Brehmer, C. Millot, C. Alberola, M. Benzhora, I. Taupier-Letage, G. Chabert d'Hieres, H. Didelle, P. Gleizon, D. Obaton, M. Crépon, C. Herbaut, G. Madec, S. Speich, J. Nihoul, J. M. Beckers, P. Brasseur, E. Deleersnijder, S. Djenidi, J. Font, A. Castellon, E. Garcia-Ladona, M. J. Lopez-Garcia, M. Manriquez, M. Maso, J. Salat, J. Tintore, S. Alonso, D. Gomis, A. Viudez, M. Astraldi, D. Bacciola, M. Borghini, F. Dell'amico, C. Galli, E. Lazzoni, G. P. Gasparini, S. Sparnocchia, and A. Harzallah, 1995: Progress from 1989 to 1992 in understanding the circulation of the Western Mediterranean Sea. Oceanol. Acta, *18*, 2, 255-271.

Guilyardi E., G. Madec, L. Terray, M. Déqué, M. Pontaud, M. Imbard, D. Stephenson, M.-A. Filiberti, D. Cariolle, P. Delecluse, and O. Thual, 1995: Simulation couplée océan-atmosphère de la variabilité du climat, C. R. Acad. Sci. Paris, t. 320, série IIa, 683-690.

Laugier M., Mortier L. and I. Dekeyser, 1995: Un modèle bidomaine emboîté aux équations primitives pour des applications océaniques. *J. Rech Océanographique*, *20*, n°3-4, 109-116.

Mechoso C., A. Robertson, N. Barth, M. Davey, P. Delecluse, P. Gent, S., Ineson, B. Kirtman, M. Latif, H. Le Treut, T. Nagai, J. D. Neelin, S. G. H. Philander, J. Polcher, P. Schopf, T. Stockdale, M. Suarez, L. Terray, O. Thual, and J. Tribbia, 1995: The seasonal cycle over the tropical Pacific in general circulation models. Mon. Wea. Rev, *123*, 9, 2825-2838.

Terray L., O. Thual, S. Belamari, M. Déqué, P. Dandin, P. Delecluse, and C. Lévy, 1995: Climatology and interannual variability simulated by the ARPEGE-OPA coupled model. Clim. Dyn., *11*, 487-505.

1996

Gavart, M., G. Caniaux, S. Planton, and H. Giordani, 1996: Évaluation d'une méthode inverse pour la détermination des courants dans le bassin des Açores. C. R. Acad. Sci. Paris, 323, série II a, 1023--1034.

Herbaut C., L. Mortier, and M. Crépon, 1996: A sensitivity study of the general circulation of the Western Mediterranean Sea. Part I: the response to density forcing through the straits. *J. Phys. Oceanogr.*, *26*, 65-84.

Laugier, M., P. Angot, and L. Mortier, 1996: Nested-grid methods for an ocean model; a comparative study. *Int. J. Numer. Meth. in Fluids*, *23*, 1163-1195.

Madec G., and M. Imbard, 1996: A global ocean mesh to overcome the North Pole singularity. *Clim. Dyn., 12*, 381-388.

Madec, G., F. Lott, P. Delecluse and M. Crépon, 1996: Large scale pre-conditioning of deep water formation in the North-western Mediterranean Sea. *J. Phys. Oceanogr, 26*, 1393-1408.

Speich S., G. Madec, and M. Crépon, 1996: The circulation in the Alboran Sea: a sensitivity study. *J. Phys. Oceanogr., 26, 3*, 320-340.

1997

Arnault S., N. Ferry, M. P. Gauthier, J. Poitevin, and H. Roquet, 1997: Altimétrie TOPEX-POSEIDON face à un modèle de circulation générale océanique en Atlantique tropical. AVISO Altimetry Newsletters, *5*, 21-23.

Blanke B., and S. Raynaud, 1997: Kinematics of the Pacific Equatorial Undercurrent: an Eulerian and Lagrangian approach from GCM results. *J. Phys. Oceanogr.*, *27*, 1038-1053.

Boulanger, J.-P., P. Delecluse, C. Maes, and C. Lévy, 1997: Long equatorial waves in a high-resolution OGCM simulation of the tropical Pacific ocean during the 1985-1994 TOGA period. *Mon. Wea. Rev., 125*, 972-984.

Braconnot, P., O. Marti, and S. Joussaume, 1997: Adjustment and feedbacks in a global coupled ocean-atmosphere model , *Clim. Dyn., 13*, 507-519.

Guilyardi E., and G. Madec, 1997: Performance of the OPA/ARPEGE-T21 global ocean-atmosphere coupled model. *Clim. Dyn., 13*, 149-165.

Herbaut C., F. Martel, and M. Crépon, 1997: A sensitivity study of the general circulation of the Western Mediterranean Sea. Part II: the response to atmospheric forcing. *J. Phys. Oceanogr.*, *27*, 2126-2144.

Madec, G. and P. Delecluse, 1997: The OPA/ARPEGE and OPA/LMD Global Ocean-Atmosphere Coupled Model. *Int. WOCE Newsletter*, *26*, 12-15.

Maes C., G. Madec and P. Delecluse, 1997: Sensitivity of an Equatorial Pacific OGCM to the lateral diffusion. *Mon. Wea. Rev.*, *125*, 5, 958-971.

Macias, J., and D. B. Stephenson, 1997: Seasonal and interannual ENSO variability in a Hybrid Coupled Model of the tropical Pacific. *C. R. Acad. Sci Paris*, Earth & Planetary Sciences, *324*, 269-276.

Vintzileos, A., and R. Sadourny, 1997: A general interface between an atmospheric general circulation model and underlying ocean and land surface models: delocalized physics scheme. *Mon. Wea. Rev.*, *125*, 926-941.

1998

Aumont, O., J. C. Orr, D. Jamous, P. Monfray, O. Marti, and G. Madec, 1998: A degradation approach to accelerate simulations to steady state in a 3-D tracer transport model of the global ocean. *Clim. Dyn.*,*14*, 101-116.

Aumont, O., P. Monfray, J. C. Orr, and G. Madec, 1998: Nutrient trapping in the equatorial Pacific: The ocean circulation solution. *Note du Pôle de modélisation*, Institut Pierre-Simon Laplace, *N°5*, 35pp.

Barthelet P., L. Terray, and S. Valcke, 1998: Transient CO_2 experiment using the ARPEGE/OPAICE non-flux corrected coupled model. *Geophys. Res. Let.*, *25*, 2277-2280.

Barthelet, P., S. Bony, P. Braconnot, A. Braun, D. Cariolle, E. Cohen-Solal, J.-L. Dufresne, P. Delecluse, M. Déqué, L. Fairhead, M.-A. Filiberti, M. Forichon, J.-Y. Grandpeix, E. Guilyardi, M.-N. Houssais, M. Imbard, H. Le Treut, C. Levy, Z. X. Li, G. Madec, P. Marquet, O. Marti, S. Planton, L. Terray, O. Thual, S, Valcke, 1998: Simulations couplées globales des changements climatiques associes a une augmentation de la teneur atmosphériques en CO_2. *C. R. Acad. Sci Paris*, Earth & Planetary Sciences, *326*, 677-684.

Caniaux, G., and S. Planton, 1998: A three-dimensional ocean mesoscale simulation using data from the SEMAPHORE experiment: mixed layer heat budget. *J. Geophys. Res.*, *103*, 25,081-25,099.

Cassou, C. P. Noyret, E. Sevault, O. Thual, L. Terray, D. Beaucourt, and M. Imbard, 1998: Distributed Ocean-Atmosphere modeling and sensitivity to the coupling flux precision: the CATHODe project. *Mon. Wea. Rev., 126,* 1035-1053.

Delecluse P., M. Davey, Y. Kitamura, S.G.H. Philander, M. Suarez, and L. Bengtsson, 1998: TOGA Review Paper: Coupled general circulation modeling of the tropical Pacific. *J. Geophys. Res.*, 14,357-14,373.

Greiner, E., S. Arnault, and A. Morlière, 1998a: Twelve monthly experiments of 4D-variational assimilation in the tropical Atlantic during 1987. Part 1: Method and statistical results. *Prog. Oceanogr.*, *41*, 141-202.

Greiner, E., S. Arnault, and A. Morlière, 1998b: Twelve monthly experiments of 4D-variational assimilation in the tropical Atlantic during 1987. Part 2: Oceanographic interpretation. *Prog. Oceanogr.*, *41*, 203-247.

Herbaut, C., F. Codron, and M. Crépon, 1998: Separation of a coastal current at a strait level: case of the strait of Sicily. *J. Phys. Oceanogr.*, *28*, 1346-1362.

Jézéquel F. et Chesneaux J.M., 1998: Etude de la stabilite numerique du code OPA - version 8.0 . *Note technique du Pole de modelisation* Institut Pierre-Simon Laplace, Paris, France.

Laurent, C., H. Le Treut, Z. X. Li, L. Fairhead, and J.-L. Dufresne, 1998: The influence of resolution in simulating inter-annual and inter-decadal variability in a coupled ocean-atmosphere GCM, with emphasis over the North Atlantic. *Note du Pôle de modélisation*, Institut Pierre-Simon Laplace, *N°8*, 38pp.

Lévy, M., L. Mémery, and G. Madec, 1998: The onset of a bloom after deep winter convection in the Northwestern Mediterranean Sea: mesoscale process study with a primitive equation model. *J. Mar. Syst.*, *16/1-2*, 7-21.

Lévy M., L. Mémery, and G. Madec, 1998: Combined effects of mesoscale processes and atmospheric high-frequency variability on the spring bloom in the MEDOC area. *Note du Pôle de modélisation*, Institut Pierre-Simon Laplace, *N°7*, 38pp.

Maes, C., 1998: Estimating the influence of salinity on sea level anomaly in the ocean. *Geophys. Res. Let.*, *25*, 3551-3554.

Maes, C., P. Delecluse, and G. Madec, 1998: Impact of westerly wind bursts on the warm pool of the TOGA-COARE domain in an OGCM. *Climate Dyn. 14*, 55-70.

Terray, L., 1998: Sensitivity of climate drift to physical parameterizations in a coupled ocean-atmosphere general circulation model. *J. Climate*, *11*, 1633-1658.

Tusseau-Vuillemin, M.-H., L. Mortier, and C. Herbaut, 1998: Modeling nitrate fluxes in an open coastal environment (Gulf of Lions): transport versus biogeochemical processes, *J. Geophys. Res., 103*, 7693-7708.

Vialard, J., and P. Delecluse, 1998: An OGCM study for the TOGA decade. Part I: Role of salinity in the physics of the western pacific fresh pool, Part II: Barrier-layer formation and variability. *Note du Pôle de modélisation*, Institut Pierre-Simon Laplace, *N°4*, 56pp.

Vialard, J., and P. Delecluse, 1998: An OGCM study for the TOGA decade. Part I: Role of salinity in the physics of the western pacific fresh pool. *J. Phys. Oceanogr., 28*, 1071-1188.

Vialard, J., and P. Delecluse, 1998: An OGCM study for the TOGA decade. Part II: Barrier-layer formation and variability. *J. Phys. Oceanogr., 28*, 1089-1106.

1999: published, in press and in revision

Aumont, O., P. Monfray, J. C. Orr, and G. Madec, 1999: Nutrient trapping in the equatorial Pacific: The ocean circulation solution. *Global Biogeochemical Cycles*, **In press**.

Blanke, B., M. Arhan, G. Madec, and S. Roche, 1999: Warm water paths in the equatorial Atlantic as diagnosed with a general circulation model. *J. Phys. Oceanogr,* **In press**

Delecluse, P., and G. Madec, 1999: Ocean modelling and the role of the ocean in the climate system. Les Houches summer school, 1-68. **In press**

Dewitte, B., G. Reverdin, and C. Maes, 1999: Vertical structure of an OGCM: interpretation in terms of long equatorial waves and comparison with linear simulations. *J. Phys. Oceanogr.*, **in press.**

Grima, N., A. Bentamy , P. Delecluse P., K. Katsaros, C. Lévy, and Y. Quilfen, 1999: Sensibility study of an Oceanic General Circulation Model forced by satellite wind-stress fields. *J. Geophys. Res.*, **in press**

Guilyardi, E., G. Madec, and L. Terray 1999: The role of lateral ocean physics in the upper ocean thermal balance of a coupled ocean-atmosphere GCM. *J. Phys. Oceanogr.*, **In revision**.

Guyon, M., G. Madec, F.-X. Roux, C. Herbaut, M. Imbard, and P. Fraunié, 1999: Domain decomposition method as a nutshell for massively parallel ocean modelling with OPA model. *Note du Pôle de modélisation*, Institut Pierre-Simon Laplace, **In press**.

Josse, P., G. Caniaux, H. Giordani, and S. Planton, 1999: Intercomparison of ocean-atmosphere fluxes from oceanic and atmospheric forced and coupled mesoscale simulations. *Annales Geophysicae*, **In press**.

Lazar, A., and J. Rancher, 1999: Simulation of radionucleide dispersion in the Pacific Ocean from Mururoa Atoll. *J. Environ. Radioactivity*, **in press**.

Lazar, A., G. Madec, and P. Delecluse, 1999: A rationalization of the Veronis upwelling / downwelling system and its sensitivity to mixing parameterizations in an idealized OGCM. *J. Phys. Oceanogr.*, **in press**.

Lehodey P., J.-M. André, M. Bertignac, A. Stoens, C. Menkes, L. Mémery, N. Grima, 1999: Predicting skipjack forage distributions in the Equatorial Pacific: preliminary results using a coupled dynamical bio-geochemical model. *Fisheries Oceanography*, **in press**.

Lévy M., L. Mémery, and G. Madec, 1999: The onset of the spring bloom in the MEDOC area: mesoscale spatial variability. *Deep Sea Res.* , **in press**.

Lévy M., L. Mémery, and G. Madec, 1999: Combined effects of mesoscale processes and atmospheric high-frequency variability on the spring bloom in the MEDOC area. *Deep Sea Res.*, **in revision**.

Maes, C., 1999: A note on the vertical scales of temperature and salinity and their signature in dynamic height in the western Pacific Ocean. Implications for data assimilation. *J. Geophys. Res.*, **in press**.

Maes, C., M. Benkiran, and P. De Mey, 1999: Sea level comparison between Topex/Poseidon altimetric data and a global ocean general circulation model from an assimilation perspective. *J. Geophys. Res.*, **in press**.

Stoens A., C. Menkes, M.-H. Radennac, Y. Dandonneau, N. Grima, G. Eldin, L. Mémery, C. Navarette, J.-M. André, T. Moutin, and P. Raimbault, 1999: The coupled physical-new production system in the equatorial Pacific during the 1992-1995 El NINO. *J. Geophys. Res.*, **in press**.

Stoens, A., C. Menkes, Y. Dandonneau, and L. Mémery, 1999: New production in the equatorial Pacific: a coupled dynamical-biogeochemical model. *Fisheries Oceanography*, **in press**.

Vintzileos, A., P. Delecluse, and R. Sadourny, 1999: On the mechanisms in a tropical ocean-global atmosphere coupled general circulation model. Part I: mean state and seasonal cycle. *Clim. Dyn.*, *15*, 43-62.

Vintzileos A., Delecluse P. and R. Sadourny, 1999: On the mechanisms in a tropical ocean-global atmosphere coupled general circulation model. Part II: interannual variability and its relation with the seasonal cycle. *Climate Dyn. 15*, 63-80

1999: submitted and in preparation

Covey, C., A. Abe-Ouchi, G. J. Boer, G. M. Flato, B. A. Boville, G. A. Meehl, U. Cubasch, E. Roeckner, H. Gordon, E. Guilyardi, L. Terray, X. Jiang, R. Miller, G. Russell, T. C. Johns, H. Le Treut, L. Fairhead, G. Madec, A. Noda, S. B. Power, E. K. Schneider, R. J. Stouffer, and H. von Storch, 1998: The seasonal cycle in coupled ocean-atmosphere general circulation models. **In preparation**.

Fairhead, L., J.-L. Dufresne, H. Le Treut, Z. X. Li, J.-Y. Grandpeix, M. Forichon, S. Bony, P. Braconnot, O. Marti, G. Madec, M.-A. Filiberti, M.-N. Houssais, and M. Imbard, 1998: The IPSL coupled atmosphere-ocean GCM: description and climatology. **In preparation**.

Ferron, B., H. Mercier, and A.M. Tréguier, 1998: Hydraulic control in the Romanche Fracture Zone. *J. Mar. Res.*, **submitted.**

Février S., C. Frankignoul, J. Sirven, M.K. Davey, P. Delecluse, S. Ineson, J. Macias, N. Scoffier, and D. B. Stephenson, 1998: A multivariate intercomparison between three oceanic GCMs using observed current and thermocline depth anomalies in the tropical Pacific during 1985-1992. *J. Geophys. Res.*, **submitted**.

Gavart, M., P. De Mey, and G. Caniaux, 1998: Assimilation of altimeter data into a primitive equation model of the Azores-Madeira region. *Dyn. Ocean Atmos.*, **submitted**

Latif, M., K. Sperber, J. Arblaster, P. Braconnot, D. Chen, A. Colman, U. Cubasch, M. Davey, P. Delecluse, D. De Witt, L. Fairhead, G. Flato, M. Ji, M. Kimoto, A. Kitoh, T. Knutson, H. Le Treut, S. Manabe, O. Marti, C. Mechoso, G. Meehl, J. Oberhuber, S. Power, E. Roeckner, L. Terray, A. Vintzileos, R. Voβ, B. Wang, W. Washington, I. Yoshikawa, J. Yu, S. Zebiak, and others…, 1998: ENSIP: The El Niño Simulation Intercomparison Project. **In preparation**

Guyon, M., G. Madec, F.-X. Roux, M. Imbard, and P. Fronier, 1999 : Parallelization of the OPA ocean model. *Technique et Science Informatique*, **submitted**

Le Quéré, C., J. C. Orr, P. Monfray, O. Aumont and G. Madec, 1998: Interannual variability of the global and regional sea-air flux of $C0_2$ from 1979 to 1993. *Tellus*, **submitted**

Loukos, H., and L. Mémery: 1998: Nitrate budget in the upper Equatorial Atlantic Ocean. Simulation of the years 1983-84 (FOCAL/SEQUAL). *J. Geophys. Res.*, **submitted**

Roullet, G., and G. Madec, 1999 : Representing surface fresh water flux as volum flux: A new free surface formulation. **In preparation**

Speer, K., E. Guilyardi, and G. Madec, 1998: Southern ocean transformation in a coupled model with and without eddy mass fluxes. **In preparation**.

Phd THESIS

1985

Chartier, M., 1985: Un modèle numérique tridimensionnel aux équations primitives de circulation générale de l'océan. Thèse de l'Université Pierre et Marie Curie, Paris, France, 111pp.

1989

Andrich P., 1989: Développement de modèles numériques océaniques pour l'étude du couplage océan-atmosphère en zone tropicale. Thèse de l'Université Pierre et Marie Curie, Paris, France, 93pp.

1990

Madec, G., 1990: La formation d'eau profonde en Méditerranée nord occidentale, son influence sur la circulation à méso-échelle: une approche numérique. Thèse de l'université Pierre et Marie Curie, Paris, France, 175pp.

1992

Blanke B., 1992: Couche de mélange dans un modèle tropical de circulation générale océanique. Thèse de l'Université Pierre et Marie Curie, Paris, France, 181pp.

Braconnot, P., 1992: Validation objective de modèles d'océan tropical à l'aide des données FOCAL/SEQUAL. Thèse de l'Université de Pierre et Marie Curie, Paris, France, 200pp.

Marti O., 1992: Étude de l'océan mondial: modélisation de la circulation et du transport de traceurs anthropogéniques. Thèse de l'Université de Pierre et Marie Curie, Paris, France, 201pp.

Mortier L., 1992: Les instabilités du courant algérien. Thèse de l'Université Aix-Marseille II, Marseille, France, 326pp.

Speich S., 1992: Étude du forçage de la circulation générale océanique par les détroits: cas de la mer d'Alboran. Thèse de l'Université Pierre et Marie Curie, Paris, France, 245pp.

1993

Dandin, P., 1993: Variabilité basse fréquence simulée dans l'océan Pacifique tropical. Thèse de l'Université Pierre et Marie Curie, Paris, France, 230pp.

Greiner, E., 1993: Mise en œuvre de méthodes de contrôle optimal pour l'assimilation de données in situ et satellitaires dans les modèles océaniques. Thèse de l'Université Pierre et Marie Curie, Paris, France, 211pp.

1994

Boulanger, J.-P., 1994: Influence des ondes équatoriales sur la variabilité basse-fréquence de l'océan Pacifique tropical. Thèse de l'Université Pierre et Marie Curie, Paris, France, 211pp.

Herbaut, C., 1994: Étude de la circulation océanique en Méditerranée Occidentale. Thèse de l'Université Pierre et Marie Curie, Paris, France, 90pp.

1995

Guyon, M., 1995:Contribution à la modélisation océanique sur architectures parallèles par une approche de décomposition de domaine. Thèse de l'Université Aix-Marseille II, Marseille, France, 202pp.

Laugier, M., 1995: Développement d'un modèle multidomaine aux équations primitives pour des applications océaniques. Thèse de l'Université d'Aix Marseille II, Marseille, France, 232pp.

Loukos, H., 1995: Simulation du cycle océanique du carbone dans l'atlantique équatorial. Validation de l'année 1983 (FOCAL). Thèse de l'Université de Pierre et Marie Curie, Paris, France, 207pp.

1996

Gavart, M., 1996: Modélisation et assimilation de données dans un modèle de circulation océanique à méso-échelle: application à la campagne SEMAPHORE. Thèse de l'Université Paul Sabatier, Toulouse, France, 208pp.

Lévy, M., 1996: Modélisation des processus biogéochimiques en Méditerranée nord-occidentale. Cycle saisonnier et variabilité mésoéchelle. Thèse de l'Université Pierre et Marie Curie, Paris, France, 331pp.

Maes C., 1996: Équilibre du réservoir chaud de l'océan Pacifique tropical ouest. Thèse de l'Université Pierre et Marie Curie, Paris, France, 258pp.

Pontaud, M., 1996: Variabilité interannuelle dans le Pacifique tropical et instabilités couplées océan-atmosphère: développements analytiques et applications à des simulations climatiques. Thèse de l'Université Paul Sabatier, Toulouse, France, 185pp.

Vintzileos, A., 1996: Étude de la variabilité climatique avec un modèle couplé atmosphère globale - océan Pacifique tropical. Université de Pierre et Marie Curie, Paris, France, 175pp.

1997

Deleville, S., 1997: Contribution à la modélisation de la dynamique marine d'été du golfe du Lion. Application d'un modèle emboîté passif. Thèse de l'Université Aix-Marseille II, Marseille, France, 174pp.

Gervasio, L., 1997: Instabilités des courants côtiers en présence de topographie. Application au courant algérien. Thèse de l'Université Pierre et Marie Curie, Paris, France, 299pp.

Grima, N., 1997: Détermination de champs de vent et de tension satellitaux. Impacts à travers un modèle de circulation océanique dans les régions tropicales. Thèse de l'Université Pierre et Marie Curie, Paris, France, 209pp

Guilyardi, E., 1997: Rôle de la physique océanique sur la formation/consommation des masses d'eau dans un modèle couplé océan-atmosphère. Université Paul Sabatier, Toulouse, France, 195pp.

Lazar, A., 1997: La branche froide de la circulation thermohaline: sensibilité à la diffusion turbulente dans un modèle de circulation générale idéalisée. Thèse de l'Université Pierre et Marie Curie, Paris, France, 269pp.

Vialard, J., 1997: Influence de la salinité sur les interactions océan-atmosphère dans le Pacifique tropical. Université de Pierre et Marie Curie, Paris, France, 218pp.

1998

Aumont, O., 1998: Étude du cycle naturel du carbone dans un modèle 3D de l'océan mondial. Thèse de l'Université de Pierre et Marie Curie, Paris, France, 340pp.

Balle-Beganton, J., 1998: Biogeochimie de l'Atlantique Equatorial: Simulation tri-dimensionnelle du Cycle de l'Azote sur l'Année 1983. Thèse de l'Université de Pierre et Marie Curie, Paris, France, 210pp.

Bonjean, B., 1998: L'influence des courants de surface sur la temperature superficielle de l'ocean pacifique tropical. Thèse de l'Université de Pierre et Marie Curie, Paris, France, 209pp.

Dutay, J.-C., 1998: Influence du mélange vertical et de la couche mélangée sur la ventilation de l'océan. Simulations numériques des traceurs transitoires tritium-helium-3 et CFCs avec le modèle OPA. Thèse de l'Université de Pierre et Marie Curie, Paris, France, 250pp.

Ferron, B., 1998: Écoulement de l'Eau Antarctique de Fond dans la Zone de Fracture Romanche. Thèse de l'Université de Bretagne Occidentale, Brest, France, 167pp.

Stoens, A., 1998: Étude de la variabilité du système couplé physique - biogéochimique dans le Pacifique tropical entre les années 1992 et 1995. Thèse de l'Université de Pierre et Marie Curie, Paris, France, 230pp.

APPENDIX A : CURVILINEAR *s*-COORDINATE EQUATIONS

In order to establish the set of Primitive Equation in curvilinear *s*-coordinates (i.e. orthogonal curvilinear coordinates in the horizontal and *s*-coordinates in the vertical), we start from the set of equation established in § I.3 for the special case $k = z$ and thus $e_3 = 1$, and we introduce an arbitrary vertical coordinate $s = s(i, j, z)$. Let us define a new vertical scale factor by $e_3 = \partial z / \partial s$ (which now depends on (i, j, z)) and the horizontal slope of *s*-surfaces by :

$$\sigma_1 = \frac{1}{e_1} \left. \frac{\partial z}{\partial i} \right|_s , \quad \text{and} \quad \sigma_2 = \frac{1}{e_2} \left. \frac{\partial z}{\partial j} \right|_s \tag{A.1}$$

The chain rule to establish the model equations in the curvilinear s-coordinate system is:

$$\begin{aligned}
\left. \frac{\partial \bullet}{\partial i} \right|_z &= \left. \frac{\partial \bullet}{\partial i} \right|_s + \frac{\partial \bullet}{\partial s} \frac{\partial s}{\partial i} = \left. \frac{\partial \bullet}{\partial i} \right|_s - \frac{e_1}{e_3} \sigma_1 \frac{\partial \bullet}{\partial s} \\
\left. \frac{\partial \bullet}{\partial j} \right|_z &= \left. \frac{\partial \bullet}{\partial j} \right|_s + \frac{\partial \bullet}{\partial s} \frac{\partial s}{\partial j} = \left. \frac{\partial \bullet}{\partial j} \right|_s - \frac{e_2}{e_3} \sigma_2 \frac{\partial \bullet}{\partial s} \\
\frac{\partial \bullet}{\partial z} &= \frac{1}{e_3} \frac{\partial \bullet}{\partial s}
\end{aligned} \tag{A.2}$$

Using (A.2), the divergence of the velocity is transformed as follows:

$$\begin{aligned}
\nabla \cdot \mathbf{U} &= \frac{1}{e_1 e_2} \left[\left. \frac{\partial (e_2 u)}{\partial i} \right|_z + \left. \frac{\partial (e_1 v)}{\partial j} \right|_z \right] + \frac{\partial w}{\partial z} \\
&= \frac{1}{e_1 e_2} \left[\left. \frac{\partial (e_2 u)}{\partial i} \right|_s - \frac{e_1}{e_3} \sigma_1 \frac{\partial (e_2 u)}{\partial s} \right. \\
&\qquad \left. + \left. \frac{\partial (e_1 v)}{\partial j} \right|_s - \frac{e_2}{e_3} \sigma_2 \frac{\partial (e_1 v)}{\partial s} \right] + \frac{\partial w}{\partial s} \frac{\partial s}{\partial z} \\
&= \frac{1}{e_1 e_2} \left[\left. \frac{\partial (e_2 u)}{\partial i} \right|_s + \left. \frac{\partial (e_1 v)}{\partial j} \right|_s \right] + \frac{1}{e_3} \left[\frac{\partial w}{\partial s} - \sigma_1 \frac{\partial u}{\partial s} - \sigma_2 \frac{\partial v}{\partial s} \right] \\
&= \frac{1}{e_1 e_2 e_3} \left[\left. \frac{\partial (e_2 e_3 u)}{\partial i} \right|_s - e_2 u \left. \frac{\partial e_3}{\partial i} \right|_s + \left. \frac{\partial (e_1 e_3 v)}{\partial j} \right|_s - e_1 v \left. \frac{\partial e_3}{\partial j} \right|_s \right] \\
&\qquad + \frac{1}{e_3} \left[\frac{\partial w}{\partial s} - \sigma_1 \frac{\partial u}{\partial s} - \sigma_2 \frac{\partial v}{\partial s} \right]
\end{aligned}$$

Noting that $\frac{1}{e_1} \left. \frac{\partial e_3}{\partial i} \right|_s = \frac{1}{e_1} \left. \frac{\partial^2 z}{\partial i \partial s} \right|_s = \frac{\partial}{\partial s} \left(\frac{1}{e_1} \left. \frac{\partial z}{\partial i} \right|_s \right) = \frac{\partial \sigma_1}{\partial s}$ and $\frac{1}{e_2} \left. \frac{\partial e_3}{\partial j} \right|_s = \frac{\partial \sigma_2}{\partial s}$, it becomes:

$$\begin{aligned}
&= \frac{1}{e_1 e_2 e_3} \left[\left. \frac{\partial (e_2 e_3 u)}{\partial i} \right|_s + \left. \frac{\partial (e_1 e_3 v)}{\partial j} \right|_s \right] \\
&\qquad + \frac{1}{e_3} \left[\frac{\partial w}{\partial s} - u \frac{\partial \sigma_1}{\partial s} - v \frac{\partial \sigma_2}{\partial s} - \sigma_1 \frac{\partial u}{\partial s} - \sigma_2 \frac{\partial v}{\partial s} \right] \\
&= \frac{1}{e_1 e_2 e_3} \left[\left. \frac{\partial (e_2 e_3 u)}{\partial i} \right|_s + \left. \frac{\partial (e_1 e_3 v)}{\partial j} \right|_s \right] \\
&\qquad + \frac{1}{e_3} \left[\frac{\partial w}{\partial s} - \frac{\partial (u \sigma_1)}{\partial s} - \frac{\partial (v \sigma_2)}{\partial s} \right]
\end{aligned}$$

Introducing a "vertical" velocity ω as the velocity normal to *s*-surfaces:

$$\omega = w - \sigma_1 u - \sigma_2 v \tag{A.3}$$

the divergence of the velocity is given in curvilinear *s*-coordinates by:

$$\nabla \cdot \mathbf{U} = \frac{1}{e_1 e_2 e_3} \left[\left. \frac{\partial (e_2 e_3 u)}{\partial i} \right|_s + \left. \frac{\partial (e_1 e_3 v)}{\partial j} \right|_s \right] + \frac{1}{e_3} \frac{\partial \omega}{\partial s} \tag{A.4}$$

As a result, the continuity equation (I.1.3) in *s*-coordinates becomes:

$$\frac{1}{e_1 e_2 e_3} \left[\left. \frac{\partial (e_2 e_3 u)}{\partial i} \right|_s + \left. \frac{\partial (e_1 e_3 v)}{\partial j} \right|_s \right] + \frac{1}{e_3} \frac{\partial \omega}{\partial s} = 0 \tag{A.5}$$

Momentum equation:

As an example let us consider (I.3.10), the first component of the momentum equation. Its non linear term can be transformed as follows:

$$+\zeta\big|_z v-\frac{1}{2e_1}\frac{\partial(u^2+v^2)}{\partial i}\bigg|_z-w\frac{\partial u}{\partial z}$$

$$=\frac{1}{e_1e_2}\left[\frac{\partial(e_2v)}{\partial i}\bigg|_z-\frac{\partial(e_1u)}{\partial j}\bigg|_z\right]v-\frac{1}{2e_1}\frac{\partial(u^2+v^2)}{\partial i}\bigg|_z-w\frac{\partial u}{\partial z}$$

$$=\frac{1}{e_1e_2}\left[\frac{\partial(e_2v)}{\partial i}\bigg|_s-\frac{\partial(e_1u)}{\partial j}\bigg|_s\right.$$
$$\left.-\frac{e_1}{e_3}\sigma_1\frac{\partial(e_2v)}{\partial s}+\frac{e_2}{e_3}\sigma_2\frac{\partial(e_1u)}{\partial s}\right]v$$
$$-\frac{1}{2e_1}\left(\frac{\partial(u^2+v^2)}{\partial i}\bigg|_s-\frac{e_1}{e_3}\sigma_1\frac{\partial(u^2+v^2)}{\partial s}\right)-\frac{w}{e_3}\frac{\partial u}{\partial s}$$

$$=\zeta\big|_s v-\frac{1}{2e_1}\frac{\partial(u^2+v^2)}{\partial i}\bigg|_s-\frac{w}{e_3}\frac{\partial u}{\partial s}-\left[\frac{\sigma_1}{e_3}\frac{\partial v}{\partial s}-\frac{\sigma_2}{e_3}\frac{\partial u}{\partial s}\right]v$$
$$+\frac{\sigma_1}{2e_3}\frac{\partial(u^2+v^2)}{\partial s}$$

$$=\zeta\big|_s v-\frac{1}{2e_1}\frac{\partial(u^2+v^2)}{\partial i}\bigg|_s$$
$$-\frac{1}{e_3}\left[w\frac{\partial u}{\partial s}+\sigma_1v\frac{\partial v}{\partial s}-\sigma_2v\frac{\partial u}{\partial s}-\sigma_1u\frac{\partial u}{\partial s}-\sigma_1v\frac{\partial v}{\partial s}\right]$$

$$=\zeta\big|_s v-\frac{1}{2e_1}\frac{\partial(u^2+v^2)}{\partial i}\bigg|_s-\frac{1}{e_3}\omega\frac{\partial u}{\partial s} \qquad \text{(A.6)}$$

Therefore, the non-linear terms of the momentum equation have the same form in *z*- and *s*-coordinates

The pressure gradient term can be transformed as follows:

$$-\frac{1}{\rho_o e_1}\frac{\partial p}{\partial i}\bigg|_z=-\frac{1}{\rho_o e_1}\left[\frac{\partial p}{\partial i}\bigg|_s-\frac{e_1}{e_3}\sigma_1\frac{\partial p}{\partial s}\right]$$

$$=-\frac{1}{\rho_o\, e_1}\frac{\partial p}{\partial i}\bigg|_s+\frac{\sigma_1}{\rho_o\, e_3}(-g\,\rho\,e_3)$$

$$=-\frac{1}{\rho_o\, e_1}\frac{\partial p}{\partial i}\bigg|_s-\frac{g\,\rho}{\rho_o}\sigma_1 \qquad \text{(A.7)}$$

An additional term appears in (A7) which accounts for the tilt of model levels.

Tracer equation:

The tracer equation is obtained using the same calculation as for the continuity equation:

$$\frac{\partial T}{\partial t}=-\frac{1}{e_1e_2e_3}\left[\frac{\partial}{\partial i}(e_2e_3Tu)+\frac{\partial}{\partial j}(e_1e_3Tv)\right.$$
$$\left.+\frac{\partial}{\partial k}(e_1e_2T\omega)\right]+D^{lT}+D^{vT} \qquad \text{(A.8)}$$

The expression of the advection term is a straight consequence of (A.4), the expression of the 3D divergence in *s*-coordinates established above.

APPENDIX B : DIFFUSIVE OPERATORS

B.1 Horizontal/Vertical 2nd Order Tracer Diffusive Operators

In z-coordinates, the horizontal/vertical second order tracer diffusive operator is given by:

$$\begin{aligned} D^T &= D^{lT} + D^{vT} \\ &= \frac{1}{e_1 e_2}\left[\frac{\partial}{\partial i}\left(\frac{e_2}{e_1} A^{lT} \frac{\partial T}{\partial i}\Big|_z \right)\Big|_z + \frac{\partial}{\partial j}\left(\frac{e_1}{e_2} A^{lT} \frac{\partial T}{\partial j}\Big|_z \right)\Big|_z \right] \\ &\qquad + \frac{\partial}{\partial z}\left(A^{vT} \frac{\partial T}{\partial z} \right) \end{aligned} \tag{B.1}$$

In s-coordinates, we defined the slopes of s-surfaces, σ_1 and σ_2 by (A.1), the vertical/horizontal ratio of diffusive coefficient by $\varepsilon = A^{vT}/A^{lT}$. The diffusive operator is given by:

$$D^T = D^{lT} + D^{vT} = \nabla|_s \cdot \left[A^{lT}\ \Re \cdot \nabla|_s (T) \right] \tag{B.2}$$

$$\text{where } \Re = \begin{pmatrix} 1 & 0 & -\sigma_1 \\ 0 & 1 & -\sigma_2 \\ -\sigma_1 & -\sigma_2 & \varepsilon + {\sigma_1}^2 + {\sigma_2}^2 \end{pmatrix}$$

or in expended form:

$$\begin{aligned} D^T = \frac{1}{e_1 e_2 e_3}\Bigg[& e_2 e_3 A^{lT} \frac{\partial}{\partial i}\left(\frac{1}{e_1}\frac{\partial T}{\partial i}\Big|_s - \frac{\sigma_1}{e_3}\frac{\partial T}{\partial s} \right)\Big|_s \\ & + e_1 e_3 A^{lT} \frac{\partial}{\partial j}\left(\frac{1}{e_2}\frac{\partial T}{\partial j}\Big|_s - \frac{\sigma_2}{e_3}\frac{\partial T}{\partial s} \right)\Big|_s \\ & + e_1 e_2 A^{lT} \frac{\partial}{\partial s}\left(-\frac{\sigma_1}{e_1}\frac{\partial T}{\partial i}\Big|_s - \frac{\sigma_2}{e_2}\frac{\partial T}{\partial j}\Big|_s \right. \\ & \left. + \left(\varepsilon + {\sigma_1}^2 + {\sigma_2}^2\right)\frac{1}{e_3}\frac{\partial T}{\partial s} \right) \Bigg] \end{aligned} \tag{B.3}$$

Equation (B.2) (or equivalently (B.3)) is obtained from (B.1) without any additional assumption. Indeed, for the special case $k = z$ and thus $e_3 = 1$, we introduce an arbitrary vertical coordinate $s = s(i, j, z)$ as in Appendix A and use (A.1) and (A.2). Since no cross horizontal derivate $\partial_i \partial_j$ appears neither in (B.1) nor in (A.2), there is a decoupling between (i, z) and (j, z) plans as well as (i, s) and (j, s) plans. The demonstration can then be done for the $(i, s) \to (j, s)$ transformation without loss of generality:

$$\begin{aligned} D^T &= \frac{1}{e_1 e_2}\frac{\partial}{\partial i}\left(\frac{e_2}{e_1} A^{lT} \frac{\partial T}{\partial i}\Big|_z \right)\Big|_z + \frac{\partial}{\partial z}\left(A^{vT} \frac{\partial T}{\partial z} \right) \\ &= \frac{1}{e_1 e_2}\left[\frac{\partial}{\partial i}\left(\frac{e_2}{e_1} A^{lT} \left(\frac{\partial T}{\partial i}\Big|_s - \frac{e_1 \sigma_1}{e_3}\frac{\partial T}{\partial s} \right) \right)\Big|_s \right. \\ &\qquad \left. - \frac{e_1 \sigma_1}{e_3}\frac{\partial}{\partial s}\left(\frac{e_2}{e_1} A^{lT} \left(\frac{\partial T}{\partial i}\Big|_s - \frac{e_1 \sigma_1}{e_3}\frac{\partial T}{\partial s} \right)\Big|_s \right) \right] \\ &\qquad + \frac{1}{e_3}\frac{\partial}{\partial s}\left[\frac{A^{vT}}{e_3}\frac{\partial T}{\partial s} \right] \\ &= \frac{1}{e_1 e_2 e_3}\Bigg[\frac{\partial}{\partial i}\left(\frac{e_2 e_3}{e_1} A^{lT} \frac{\partial T}{\partial i}\Big|_s \right)\Big|_s - \frac{e_2}{e_1} A^{lT} \frac{\partial e_3}{\partial i}\Big|_s \frac{\partial T}{\partial i}\Big|_s \\ &\qquad - e_3 \frac{\partial}{\partial i}\left(\frac{e_2 \sigma_1}{e_3} A^{lT} \frac{\partial T}{\partial s} \right)\Big|_s - e_1 \sigma_1 \frac{\partial}{\partial s}\left(\frac{e_2}{e_1} A^{lT} \frac{\partial T}{\partial i}\Big|_s \right) \\ &\qquad - e_1 \sigma_1 \frac{\partial}{\partial s}\left(-\frac{e_2 \sigma_1}{e_3} A^{lT} \frac{\partial T}{\partial s} \right) + \frac{\partial}{\partial s}\left(\frac{e_1 e_2}{e_3} A^{vT} \frac{\partial T}{\partial s} \right) \Bigg] \end{aligned}$$

Noting that $\frac{1}{e_1}\frac{\partial e_3}{\partial i}\Big|_s = \frac{\partial \sigma_1}{\partial s}$, it becomes:

$$\begin{aligned} = \frac{1}{e_1 e_2 e_3}\Bigg[& \frac{\partial}{\partial i}\left(\frac{e_2 e_3}{e_1} A^{lT} \frac{\partial T}{\partial i}\Big|_s \right)\Big|_s - e_3 \frac{\partial}{\partial i}\left(\frac{e_2 \sigma_1}{e_3} A^{lT} \frac{\partial T}{\partial s} \right)\Big|_s \\ & - e_2 A^{lT} \frac{\partial \sigma_1}{\partial s}\frac{\partial T}{\partial i}\Big|_s - e_1 \sigma_1 \frac{\partial}{\partial s}\left(\frac{e_2}{e_1} A^{lT} \frac{\partial T}{\partial i}\Big|_s \right) \\ & + e_1 \sigma_1 \frac{\partial}{\partial s}\left(\frac{e_2 \sigma_1}{e_3} A^{lT} \frac{\partial T}{\partial s} \right) + \frac{\partial}{\partial s}\left(\frac{e_1 e_2}{e_3} A^{vT} \frac{\partial T}{\partial z} \right) \Bigg] \end{aligned}$$

$$= \frac{1}{e_1 e_2 e_3} \left[\frac{\partial}{\partial i}\left(\frac{e_2 e_3}{e_1} A^{IT} \frac{\partial T}{\partial i}\Big|_s \right)\Big|_s - \frac{\partial}{\partial i}\left(e_2 \sigma_1 A^{IT} \frac{\partial T}{\partial s} \right)\Big|_s \right.$$
$$+ \frac{e_2 \sigma_1}{e_3} A^{IT} \frac{\partial T}{\partial s} \frac{\partial e_3}{\partial i}\Big|_s - e_2 A^{IT} \frac{\partial \sigma_1}{\partial s} \frac{\partial T}{\partial i}\Big|_s$$
$$- e_2 \sigma_1 \frac{\partial}{\partial s}\left(A^{IT} \frac{\partial T}{\partial i}\Big|_s \right) + \frac{\partial}{\partial s}\left(\frac{e_1 e_2 \sigma_1^2}{e_3} A^{IT} \frac{\partial T}{\partial s} \right)$$
$$\left. - \frac{\partial (e_1 e_2 \sigma_1)}{\partial s}\left(\frac{\sigma_1}{e_3} A^{IT} \frac{\partial T}{\partial s} \right) + \frac{\partial}{\partial s}\left(\frac{e_1 e_2}{e_3} A^{vT} \frac{\partial T}{\partial s} \right) \right]$$

using the same remark as just above, it becomes:

$$= \frac{1}{e_1 e_2 e_3} \left[\frac{\partial}{\partial i}\left(\frac{e_2 e_3}{e_1} A^{IT} \frac{\partial T}{\partial i}\Big|_s \right) - e_2 \sigma_1 A^{IT} \frac{\partial T}{\partial s} \right]\Big|_s$$
$$+ \frac{e_1 e_2 \sigma_1}{e_3} A^{IT} \frac{\partial T}{\partial s} \frac{\partial \sigma_1}{\partial s} - \frac{\sigma_1}{e_3} A^{IT} \frac{\partial (e_1 e_2 \sigma_1)}{\partial s} \frac{\partial T}{\partial s}$$
$$- e_2 \left(A^{IT} \frac{\partial \sigma_1}{\partial s} \frac{\partial T}{\partial i}\Big|_s + \frac{\partial}{\partial s}\left(\sigma_1 A^{IT} \frac{\partial T}{\partial i}\Big|_s \right) - \frac{\partial \sigma_1}{\partial s} A^{IT} \frac{\partial T}{\partial i}\Big|_s \right)$$
$$\left. + \frac{\partial}{\partial s}\left(\frac{e_1 e_2 \sigma_1^2}{e_3} A^{IT} \frac{\partial T}{\partial s} + \frac{e_1 e_2}{e_3} A^{vT} \frac{\partial T}{\partial s} \right) \right]$$

Since the horizontal scale factor do not depend on the vertical coordinates, the last term of the first line and the first term of the last line cancel, while the second line reduces to a single vertical derivative, so it becomes:

$$= \frac{1}{e_1 e_2 e_3} \left[\frac{\partial}{\partial i}\left(\frac{e_2 e_3}{e_1} A^{IT} \frac{\partial T}{\partial i}\Big|_s - e_2 \sigma_1 A^{IT} \frac{\partial T}{\partial s} \right)\Big|_s \right.$$
$$\left. + \frac{\partial}{\partial s}\left(- e_2 \sigma_1 A^{IT} \frac{\partial T}{\partial i}\Big|_s + A^{IT} \frac{e_1 e_2}{e_3} (\varepsilon + \sigma_1^2) \frac{\partial T}{\partial s} \right) \right]$$

in other words:

$$D^T = \frac{1}{e_1 e_2 e_3} \begin{pmatrix} \frac{\partial (e_2 e_3 \bullet)}{\partial i}\Big|_s \\ \frac{\partial (e_1 e_2 \bullet)}{\partial s} \end{pmatrix} \cdot \left[A^{IT} \begin{pmatrix} 1 & -\sigma_1 \\ -\sigma_1 & \varepsilon + \sigma_1^2 \end{pmatrix} \cdot \begin{pmatrix} \frac{1}{e_1} \frac{\partial \bullet}{\partial i}\Big|_s \\ \frac{1}{e_3} \frac{\partial \bullet}{\partial s} \end{pmatrix} (T) \right]$$

B.2 Isopycnal/Vertical Second Order Tracer Diffusive Operators

The isopycnal diffusive tensor $\mathbf{A_I}$ expressed in the curvilinear coordinate system (i, j, k), in which the equations of the ocean circulation model are formulated, takes the following expression [Redi 1982]:

$$\mathbf{A_I} = \frac{A^{IT}}{(1 + a_1^2 + a_2^2)} \begin{bmatrix} 1 + a_1^2 & -a_1 a_2 & -a_1 \\ -a_1 a_2 & 1 + a_2^2 & -a_2 \\ -a_1 & -a_2 & \varepsilon + a_1^2 + a_2^2 \end{bmatrix}$$

where (a_1, a_2) are the isopycnal slopes in $(\mathbf{i}, \mathbf{j})$ directions:

$$a_1 = \frac{e_3}{e_1} \left(\frac{\partial \rho}{\partial i} \right) \left(\frac{\partial \rho}{\partial k} \right)^{-1}$$

$$a_2 = \frac{e_3}{e_2} \left(\frac{\partial \rho}{\partial j} \right) \left(\frac{\partial \rho}{\partial k} \right)^{-1}$$

In practice, the isopycnal slopes are generally less than 10^{-2} in the ocean, so $\mathbf{A_I}$ can be simplified appreciably (Cox, 1987) :

$$\mathbf{A_I} = A^{IT} \begin{bmatrix} 1 & 0 & -a_1 \\ 0 & 1 & -a_2 \\ -a_1 & -a_2 & \varepsilon + a_1^2 + a_2^2 \end{bmatrix}$$

The resulting isopycnal operator conserves the quantity it diffuses, and dissipates the square of this quantity. The demonstration of the first property is trivial. Let us demonstrate the second one:

$$\iiint_D T \, \nabla . (\mathbf{A_I} \nabla T) \, dv = - \iiint_D \nabla T \, . (\mathbf{A_I} \nabla T) \, dv$$

since

$$\nabla T . (\mathbf{A_I} \nabla T) = A^{IT} \left[\left(\frac{\partial T}{\partial i} \right)^2 - 2 a_1 \frac{\partial T}{\partial i} \frac{\partial T}{\partial k} + \left(\frac{\partial T}{\partial j} \right)^2 \right.$$
$$\left. - 2 a_2 \frac{\partial T}{\partial j} \frac{\partial T}{\partial k} + (a_1^2 + a_2^2) \left(\frac{\partial T}{\partial k} \right)^2 \right]$$

$$= A_h \left[\left(\frac{\partial T}{\partial i} - a_1 \frac{\partial T}{\partial k} \right)^2 + \left(\frac{\partial T}{\partial j} - a_2 \frac{\partial T}{\partial k} \right)^2 \right] \geq 0$$

the property becomes obvious.

Note that the resulting tensor is similar to those obtained for geopotential diffusion in s-coordinates. The simplification leads to a decoupling between (i, z) and (j, z) plans.

The resulting diffusive operator in z-coordinates has the following expression :

$$D^T = \frac{1}{e_1 e_2} \left\{ \frac{\partial}{\partial i} \left[A_h \left(\frac{e_2}{e_1} \frac{\partial T}{\partial i} - a_1 \frac{e_2}{e_3} \frac{\partial T}{\partial k} \right) \right] \right.$$
$$\left. + \frac{\partial}{\partial j} \left[A_h \left(\frac{e_1}{e_2} \frac{\partial T}{\partial j} - a_2 \frac{e_1}{e_3} \frac{\partial T}{\partial k} \right) \right] \right\}$$
$$+ \frac{1}{e_3} \frac{\partial}{\partial k} \left[A_h \left(- \frac{a_1}{e_1} \frac{\partial T}{\partial i} - \frac{a_2}{e_2} \frac{\partial T}{\partial j} + \frac{(a_1^2 + a_2^2)}{e_3} \frac{\partial T}{\partial k} \right) \right]$$

B.3 Lateral/Vertical Momentum Diffusive Operators

* lateral/vertical 2nd order momentum diffusive operator

Following (I.3.6), the Laplacian of the horizontal velocity can be expressed in *z*-coordinates:

$$\Delta \mathbf{U}_h = \nabla\left(\nabla \cdot \mathbf{U}_h\right) - \nabla \times \left(\nabla \times \mathbf{U}_h\right)$$

$$= \begin{pmatrix} \frac{1}{e_1}\frac{\partial \chi}{\partial i} \\ \frac{1}{e_2}\frac{\partial \chi}{\partial j} \\ \frac{1}{e_3}\frac{\partial \chi}{\partial k} \end{pmatrix} - \begin{pmatrix} \frac{1}{e_2}\frac{\partial \zeta}{\partial j} - \frac{1}{e_3}\frac{\partial}{\partial k}\left(\frac{1}{e_3}\frac{\partial u}{\partial k}\right) \\ \frac{1}{e_3}\frac{\partial}{\partial k}\left(-\frac{1}{e_3}\frac{\partial v}{\partial k}\right) - \frac{1}{e_1}\frac{\partial \zeta}{\partial i} \\ \frac{1}{e_1 e_2}\left[\frac{\partial}{\partial i}\left(\frac{e_2}{e_3}\frac{\partial u}{\partial k}\right) - \frac{\partial}{\partial j}\left(-\frac{e_1}{e_3}\frac{\partial v}{\partial k}\right)\right] \end{pmatrix}$$

$$= \begin{pmatrix} \frac{1}{e_1}\frac{\partial \chi}{\partial i} - \frac{1}{e_2}\frac{\partial \zeta}{\partial j} \\ \frac{1}{e_2}\frac{\partial \chi}{\partial j} + \frac{1}{e_1}\frac{\partial \zeta}{\partial i} \\ 0 \end{pmatrix} + \frac{1}{e_3}\begin{pmatrix} \frac{\partial}{\partial k}\left(\frac{1}{e_3}\frac{\partial u}{\partial k}\right) \\ \frac{\partial}{\partial k}\left(\frac{1}{e_3}\frac{\partial v}{\partial k}\right) \\ \frac{\partial \chi}{\partial k} - \frac{1}{e_1 e_2}\left(\frac{\partial^2\left(e_2 u\right)}{\partial i \partial k} + \frac{\partial^2\left(e_1 v\right)}{\partial j \partial k}\right) \end{pmatrix}$$

Using (I.3.8), the definition of the horizontal divergence, the third componant of the second vector is obviously zero and thus :

$$\Delta \mathbf{U}_h = \nabla_h\left(\chi\right) - \nabla_h \times \left(\zeta\right) + \frac{1}{e_3}\frac{\partial}{\partial k}\left(\frac{1}{e_3}\frac{\partial \mathbf{U}_h}{\partial k}\right)$$

The lateral/vertical second order (Laplacian type) operator used to diffuse horizontal momentum in *z*-coordinates therefore takes the following expression :

$$\mathbf{D}^{\mathbf{U}} = \mathbf{D}^{lm} + \mathbf{D}^{vm}$$

$$= \nabla_h\left(A^{lm}\chi\right) - \nabla_h \times \left(A^{lm}\zeta \mathbf{k}\right) + \frac{1}{e_3}\frac{\partial}{\partial k}\left(\frac{A^{vm}}{e_3}\frac{\partial \mathbf{U}_h}{\partial k}\right)$$

$$\mathbf{D}^{\mathbf{U}} = \begin{pmatrix} \frac{1}{e_1}\frac{\partial\left(A^{lm}\chi\right)}{\partial i} - \frac{1}{e_2}\frac{\partial\left(A^{lm}\zeta\right)}{\partial j} \\ \frac{1}{e_2}\frac{\partial\left(A^{lm}\chi\right)}{\partial j} + \frac{1}{e_1}\frac{\partial\left(A^{lm}\zeta\right)}{\partial i} \end{pmatrix} + \frac{1}{e_3}\frac{\partial}{\partial k}\begin{pmatrix} \frac{A^{vm}}{e_3}\frac{\partial u}{\partial k} \\ \frac{A^{vm}}{e_3}\frac{\partial v}{\partial k} \end{pmatrix}$$

References

Cox, M. D., 1987 : Isopycnal diffusion in a z-coordinate ocean model. *Ocean Modelling*, 74, 1-9.

Redi, M. H., 1982 : oceanic isopycnal mixing by coordinate rotation. *J. Phys. Oceanogr., 13*, 1154-1158.

APPENDIX C : DISCRETE INVARIANTS OF THE EQUATIONS

C.1 Conservation Properties on Ocean Dynamics

First, the boundary condition on the vertical velocity (no flux through the surface and the bottom) is established for the discrete set of momentum equations. Then, it is shown that the non linear terms of the momentum equation are written such that the potential enstrophy of a horizontally non divergent flow is preserved while all the other non-diffusive terms preserve the kinetic energy: the energy is also preserved in practice. In addition, an option is also offer for the vorticity term discretization which provides a total kinetic energy conserving discretization for that term. Note that although these properties are established in the curvilinear *s*-coordinate system, they still hold in the curvilinear *z*-coordinate system.

C.1-a Bottom Boundary Condition on Vertical Velocity Field

The discrete set of momentum equations used in rigid lid approximation automatically satisfies the surface and bottom boundary conditions ($w_{surface}=w_{bottom}=0$, no flux through the surface and the bottom:). Indeed, taking the discrete horizontal divergence of the vertical sum of the horizontal momentum equations (Eqs. (II.2.1) and (II.2.2)) wheighted by the vertical scale factors, it becomes:

$$\frac{\partial}{\partial t}\left(\sum_k \chi\right) \equiv \frac{\partial}{\partial t}\left(w_{surface} - w_{bottom}\right)$$

$$\equiv \frac{1}{e_{1T}\, e_{2T}\, e_{3T}} \left\{ \delta_i \left[e_{2u} H_u \left(M_u - M_u - \frac{1}{H_u e_{2u}} \delta_j \left[\partial_t \psi \right] \right) \right] + \delta_j \left[e_{1v} H_v \left(M_v - M_v + \frac{1}{H_v e_{1v}} \delta_i \left[\partial_t \psi \right] \right) \right] \right\}$$

$$\equiv \frac{1}{e_{1T}\, e_{2T}\, e_{3T}} \left\{ -\delta_i \left[\delta_j \left[\partial_t \psi \right] \right] + \delta_j \left[\delta_i \left[\partial_t \psi \right] \right] \right\} \equiv 0$$

The surface boundary condition associated with the rigid lid approximation ($w_{surface}=0$) is imposed in the computation of the vertical velocity (II.2.5). Therefore, it turns out to be:

$$\frac{\partial}{\partial t} w_{bottom} \equiv 0$$

As the bottom velocity is initially set to zero, it remains zero all the time. Symmetrically, if $w_{bottom}=0$ is used in the computation of the vertical velocity (upward integral of the horizontal divergence), the same computation leads to $w_{surface}=0$ as soon as the surface vertical velocity is initially set to zero.

C.1-b Vorticity Term

Potential vorticity is located at *f*-points and defined as: ζ / e_{3f} . The standard discrete formulation of the relative vorticity term obviously conserves potential vorticity. It also conserves the potential enstrophy for a horizontally non-divergent flow (i.e. $\chi=0$) but not the total kinetic energy. Indeed, using the symmetry or skew symmetry properties of the operators (Eqs (II.1.10) and (II.1.11)), it can be shown that:

$$\int_D \zeta / e_3 \;\; \mathbf{k} \cdot \frac{1}{e_3} \nabla \times \left(\zeta\, \mathbf{k} \times \mathbf{U}_h \right) dv \equiv 0 \qquad \text{(C.1.1)}$$

where $dv = e_1\, e_2\, e_3\, di\, dj\, dk$ is the volume element. Indeed, using (II.2.11), the discrete form of the right hand side of (C.1.1) can be transformed as follow:

$$\sum_{i,j,k} \frac{\zeta / e_{3f}}{e_{1f} e_{2f} e_{3f}} \left\{ \delta_{i+1/2} \left[-\overline{\left(\zeta / e_{3f} \right)}^{\,i} \overline{\overline{\left(e_{2u} e_{3u} u \right)}}^{\,i,j+1/2} \right] - \delta_{j+1/2} \left[\overline{\left(\zeta / e_{3f} \right)}^{\,j} \overline{\overline{\left(e_{1v} e_{3v} v \right)}}^{\,i+1/2,j} \right] \right\} e_{1f} e_{2f} e_{3f}$$

$$\equiv \sum_{i,j,k} \left\{ \delta_i \left[\zeta / e_{3f} \right] \overline{\left(\zeta / e_{3f} \right)}^{\,i} \overline{\overline{\left(e_{1u} e_{3u} u \right)}}^{\,i,j+1/2} + \delta_j \left[\zeta / e_{3f} \right] \overline{\left(\zeta / e_{3f} \right)}^{\,j} \overline{\overline{\left(e_{2v} e_{3v} v \right)}}^{\,i+1/2,j} \right\}$$

$$\equiv \sum_{i,j,k} \frac{1}{2}\Big\{ \quad \delta_i\Big[\big(\zeta/e_{3f}\big)^2\Big]\ \overline{\overline{\big(e_{2u}e_{3u}u\big)}}^{\,i,j+1/2} + \delta_j\Big[\big(\zeta/e_{3f}\big)^2\Big]\ \overline{\overline{\big(e_{1v}e_{3v}v\big)}}^{\,i+1/2,j}\Big\}$$

$$\equiv \sum_{i,j,k} -\frac{1}{2}\big(\zeta/e_{3f}\big)^2\Big\{ \quad \delta_{i+1/2}\Big[\overline{\overline{\big(e_{2u}e_{3u}u\big)}}^{\,i,j+1/2}\Big] + \delta_{j+1/2}\Big[\overline{\overline{\big(e_{1v}e_{3v}v\big)}}^{\,i+1/2,j}\Big]\ \Big\}$$

Since $\overline{\cdot}$ and δ operators commute: $\delta_{i+1/2}\big[\overline{a}^{\,i}\big] = \overline{\delta_i[a]}^{\,i+1/2}$, and introducing the horizontal divergence χ, it becomes:

$$\equiv \sum_{i,j,k} -\frac{1}{2}\big(\zeta/e_{3f}\big)^2\ \overline{\overline{e_{1T}e_{2T}e_{3T}\,\chi}}^{\,i+1/2,j+1/2} \equiv 0$$

Note that the demonstration is done here for the relative potential vorticity but it still hold for the planetary (f/e_3) and the total potential vorticity ($(\zeta+f)/e_3$). Another formulation of the two components of the vorticity term is optionally offered :

$$\frac{1}{e_3}\nabla\times\big(\zeta\,\mathbf{k}\times\mathbf{U}_h\big) \equiv \begin{pmatrix} +\dfrac{1}{e_{1u}}\overline{\big(\zeta/e_{3f}\big)\overline{\big(e_{1v}e_{3v}v\big)}^{\,i+1/2}}^{\,j} \\ -\dfrac{1}{e_{2v}}\overline{\big(\zeta/e_{3f}\big)\overline{\big(e_{2u}e_{3u}u\big)}^{\,j+1/2}}^{\,i} \end{pmatrix}$$

This formulation does not conserve the enstrophy but the total kinetic energy. It is also possible to mix the two formulations in order to conserve enstrophy on the relative vorticity term and energy on the Coriolis term.

$$\int_D \mathbf{U}_h\times\big(\zeta\,\mathbf{k}\times\mathbf{U}_h\big)\,dv$$

$$\equiv \sum_{i,j,k}\Big\{\ \overline{\big(\zeta/e_{3f}\big)\overline{\big(e_{1v}e_{3v}\,v\big)}^{\,i+1/2}}^{\,j}\ e_{2u}e_{3u}\,u - \overline{\big(\zeta/e_{3f}\big)\overline{\big(e_{2u}e_{3u}\,u\big)}^{\,j+1/2}}^{\,i}\ e_{1v}e_{3v}\,v\ \Big\}$$

$$\equiv \sum_{i,j,k}\big(\zeta/e_{3f}\big)\Big\{\ \overline{\big(e_{1v}e_{3v}\,v\big)}^{\,i+1/2}\ \overline{\big(e_{2u}e_{3u}\,u\big)}^{\,j+1/2} - \overline{\big(e_{2u}e_{3u}\,u\big)}^{\,j+1/2}\ \overline{\big(e_{1v}e_{3v}\,v\big)}^{\,i+1/2}\ \Big\} \equiv 0$$

C.1-c Gradient of Kinetic Energy / Vertical Advection

The change of Kinetic Energy (KE) due to the vertical advection is exactly balanced by the change of KE due to the horizontal gradient of KE :

$$\int_D \mathbf{U}_h\cdot\nabla_h\big(1/2\,\mathbf{U}_h^{\,2}\big)\,dv = -\int_D \mathbf{U}_h\cdot w\,\frac{\partial\mathbf{U}_h}{\partial k}\,dv$$

Indeed,

$$\int_D \mathbf{U}_h\cdot\nabla_h\big(1/2\,\mathbf{U}_h^{\,2}\big)\,dv$$

$$\equiv \sum_{i,j,k}\frac{1}{2}\Big\{\ \frac{1}{e_{1u}}\delta_{i+1/2}\Big[\overline{u^2}^{\,i}+\overline{v^2}^{\,j}\Big]\,u\,e_{1u}e_{2u}e_{3u} + \frac{1}{e_{2v}}\delta_{j+1/2}\Big[\overline{u^2}^{\,i}+\overline{v^2}^{\,j}\Big]\,v\,e_{1v}e_{2v}e_{3v}\ \Big\}$$

Using (II.1.10) and the continuity of mass (II.2.5), it becomes :

$$\equiv \sum_{i,j,k}\frac{1}{2}\Big(\overline{u^2}^{\,i}+\overline{v^2}^{\,j}\Big)\,\delta_k\big[e_{1T}e_{2T}\,w\big]$$

Using (II.1.10), and the commutativity of operators $\overline{\cdot}$ and δ, it becomes :

$$\equiv -\sum_{i,j,k}\frac{1}{2}\,\delta_{k+1/2}\Big[\overline{u^2}^{\,i}+\overline{v^2}^{\,j}\Big]\,e_{1T}e_{2T}\,w$$

$$\equiv \sum_{i,j,k}\frac{1}{2}\Big(\overline{\delta_{k+1/2}\big[u^2\big]}^{\,i}+\overline{\delta_{k+1/2}\big[v^2\big]}^{\,j}\Big)\,e_{1T}e_{2T}\,w$$

Using (II.1.11) successively in the horizontal and in the vertical, it becomes :

$$\equiv \sum_{i,j,k}\frac{1}{2}\Big\{\ \overline{e_{1T}e_{2T}\,w}^{\,i+1/2}\ 2\overline{u}^{\,k+1/2}\delta_{k+1/2}[u] + \overline{e_{1T}e_{2T}\,w}^{\,j+1/2}\ 2\overline{v}^{\,k+1/2}\delta_{k+1/2}[v]\ \Big\}$$

$$\equiv -\sum_{i,j,k}\Big\{\ \frac{1}{b_u}\overline{\Big\{\overline{e_{1T}e_{2T}\,w}^{\,i+1/2}\delta_{k+1/2}[u]\Big\}}^{\,k}\,u\,e_{1u}e_{2u}e_{3u} + \frac{1}{b_v}\overline{\Big\{\overline{e_{1T}e_{2T}\,w}^{\,j+1/2}\delta_{k+1/2}[v]\Big\}}^{\,k}\,v\,e_{1v}e_{2v}e_{3v}\ \Big\}$$

$$\equiv -\int_D \mathbf{U}_h\cdot w\,\frac{\partial\mathbf{U}_h}{\partial k}\,dv$$

The main point here is that the respect of this property links the choice of the discrete formulation of vertical advection and of horizontal gradient of KE. Choosing one imposes the other. For example KE can also be defined as $1/2(\overline{u}^{\,i^2}+\overline{v}^{\,j^2})$, but this leads to the following expression for the vertical advection :

$$\frac{1}{e_3}w\frac{\partial\mathbf{U}_h}{\partial k} \equiv \begin{pmatrix} \dfrac{1}{e_{1u}e_{2u}e_{3u}}\overline{\overline{e_{1T}e_{2T}\,w\,\delta_{k+1/2}\big[\overline{u}^{\,i+1/2}\big]}}^{\,i+1/2,k} \\ \dfrac{1}{e_{1v}e_{2v}e_{3v}}\overline{\overline{e_{1T}e_{2T}\,w\,\delta_{k+1/2}\big[\overline{v}^{\,j+1/2}\big]}}^{\,j+1/2,k} \end{pmatrix}$$

This formulation requires one more horizontal mean, and thus the use of 9 velocity points instead of 3. This is the reason why it has not been retained in the model.

C.1-d Hydrostatic Pressure Gradient Term

A pressure gradient has no contribution to the evolution of the vorticity as the curl of a gradient is zero. In z-coordinates, this properties is satisfied locally with the choice of discretization we have made (property (II.1.9)). When the equation of state is linear (i.e. when a advective-diffusive equation for density can be derived from those of temperature and salinity) the change of KE due to the work of pressure forces is balanced by the change of potential energy due to buoyancy forces.

$$\int_D -\frac{1}{\rho_o}\nabla p^h\big|_z \cdot \mathbf{U}_h \; dv = \int_D \nabla.\big(\rho\,\mathbf{U}\big)\, g\, z \;\; dv$$

This property is satisfied in both z- and s-coordinates. Indeed, defining the depth of a T-point, z_T defined as the sum of the vertical scale factors at w-points starting from the surface, it can be written as:

$$\int_D -\frac{1}{\rho_o}\nabla p^h\big|_z \cdot \mathbf{U}_h \; dv$$

$$\equiv \sum_{i,j,k}\Bigg\{ -\frac{1}{\rho_o e_{1u}}\Big(\delta_{i+1/2}\big[p^h\big]-g\overline{\rho}^{\,i+1/2}\,\delta_{i+1/2}\big[z_T\big]\Big)u\,e_{1u}e_{2u}e_{3u} - \frac{1}{\rho_o e_{2v}}\Big(\delta_{j+1/2}\big[p^h\big]-g\overline{\rho}^{\,j+1/2}\delta_{j+1/2}\big[z_T\big]\Big)v\,e_{1v}e_{2v}e_{3v}\Bigg\}$$

Using (II.1.10), the continuity equation (II.2.5), and the hydrostatic equation (II.2.4), it turns out to be:

$$\equiv \sum_{i,j,k}\frac{1}{\rho_o}\Big\{ e_{2u}e_{3u}\,u\,g\overline{\rho}^{\,i+1/2}\,\delta_{i+1/2}\big[z_T\big]+e_{1v}e_{3v}\,v\,g\overline{\rho}^{\,j+1/2}\delta_{j+1/2}\big[z_T\big] +\Big(\delta_i\big[e_{2u}e_{3u}u\big]+\delta_j\big[e_{1v}e_{3v}v\big]\Big)p^h\Big\}$$

$$\equiv \sum_{i,j,k}\frac{1}{\rho_o}\Big\{ e_{2u}e_{3u}\,u\,g\overline{\rho}^{\,i+1/2}\,\delta_{i+1/2}\big[z_T\big]+e_{1v}e_{3v}\,v\,g\overline{\rho}^{\,j+1/2}\delta_{j+1/2}\big[z_T\big] -\delta_k\big[e_{1w}e_{2w}w\big]p^h\Big\}$$

$$\equiv \sum_{i,j,k}\frac{1}{\rho_o}\Big\{ e_{2u}e_{3u}\,u\,g\overline{\rho}^{\,i+1/2}\,\delta_{i+1/2}\big[z_T\big]+e_{1v}e_{3v}\,v\,g\overline{\rho}^{\,j+1/2}\delta_{j+1/2}\big[z_T\big] +e_{1w}e_{2w}\,w\,\delta_{k+1/2}\big[p_h\big]\Big\}$$

$$\equiv \sum_{i,j,k}\frac{g}{\rho_o}\Big\{ e_{2u}e_{3u}\,u\overline{\rho}^{\,i+1/2}\,\delta_{i+1/2}\big[z_T\big]+e_{1v}e_{3v}\,v\overline{\rho}^{\,j+1/2}\delta_{j+1/2}\big[z_T\big] -e_{1w}e_{2w}\,w\,e_{3w}\overline{\rho}^{\,k+1/2}\Big\}$$

noting that by definition of z_T, $\delta_{k+1/2}\big[z_T\big]\equiv -e_{3w}$, thus:

$$\equiv \sum_{i,j,k}\frac{g}{\rho_o}\Big\{ e_{2u}e_{3u}\,u\overline{\rho}^{\,i+1/2}\,\delta_{i+1/2}\big[z_T\big]+e_{1v}e_{3v}\,v\overline{\rho}^{\,j+1/2}\delta_{j+1/2}\big[z_T\big] +e_{1w}e_{2w}\,w\overline{\rho}^{\,k+1/2}\delta_{k+1/2}\big[z_T\big]\Big\}$$

Using (II.1.10), it becomes :

$$\equiv -\frac{g}{\rho_o}\sum_{i,j,k} z_T\Big\{\,\delta_i\Big[e_{2u}e_{3u}u\,\overline{\rho}^{\,i+1/2}\Big]+\delta_j\Big[e_{1v}e_{3v}v\,\overline{\rho}^{\,j+1/2}\Big] +\delta_k\Big[e_{1w}e_{2w}w\,\overline{\rho}^{\,k+1/2}\Big]\Big\}$$

$$\equiv -\int_D \nabla.\big(\rho\,\mathbf{U}\big)\, g\, z \;\; dv$$

Note that this property strongly constraints the discrete expression of both the depth of T-points and of the term added to the pressure gradient in s-coordinates.

C.1-e Surface Pressure Gradient Term

The surface pressure gradient has no contribution to the evolution of the vorticity. This property is trivially satisfied locally as the equation verified by ψ has been derived from the discrete formulation of the momentum equation and of the curl. But it has to be noticed that since the elliptic equation verified by ψ is solved numerically by an iterative solver (preconditioned conjugate gradient or successive over relaxation), the property is only satisfied at the precision required on the solver used.

With the rigid-lid approximation, the change of KE due to the work of surface pressure forces is exactly zero. This is satisfied in discrete form, at the precision required on the elliptic solver used to solve this equation. This can be demonstrated as follows:

$$\int_D -\frac{1}{\rho_o}\nabla_h\big(p_s\big)\cdot\mathbf{U}_h \; dv$$

$$\equiv \sum_{i,j,k}\Bigg\{\Big(-M_u-\frac{1}{H_u e_{2u}}\delta_j\big[\partial_t\psi\big]\Big)u\,e_{1u}e_{2u}e_{3u} +\Big(-M_v+\frac{1}{H_v e_{1v}}\delta_i\big[\partial_t\psi\big]\Big)v\,e_{1v}e_{2v}e_{3v}\Bigg\}$$

$$\equiv \sum_{i,j}\Bigg\{\Big(-M_u-\frac{1}{H_u e_{2u}}\delta_j\big[\partial_t\psi\big]\Big)\Big(\sum_k u\,e_{3u}\Big)e_{1u}e_{2u} +\Big(-M_v+\frac{1}{H_v e_{1v}}\delta_i\big[\partial_t\psi\big]\Big)\Big(\sum_k v\,e_{3v}\Big)e_{1v}e_{2v}\Bigg\}$$

using the relation between ψ and the vertically sum of the velocity, it becomes :

$$\equiv \sum_{i,j}\Bigg\{\Big(M_u+\frac{1}{H_u e_{2u}}\delta_j\big[\partial_t\psi\big]\Big)e_{1u}\,\delta_j\big[\partial_t\psi\big] +\Big(-M_v+\frac{1}{H_v e_{1v}}\delta_i\big[\partial_t\psi\big]\Big)e_{2v}\,\delta_i\big[\partial_t\psi\big]\Bigg\}$$

applying the adjoint of the δ operator, it is now:

$$\equiv \sum_{i,j} -\partial_t \psi \Big\{ \delta_{j+1/2}\left[e_{1u} M_u\right] - \delta_{i+1/2}\left[e_{1v} M_v\right] + \delta_{i+1/2}\left[\frac{e_{2v}}{H_v e_{2v}} \delta_i\left[\partial_t \psi\right]\right] + \delta_{j+1/2}\left[\frac{e_{1u}}{H_u e_{2u}} \delta_j\left[\partial_t \psi\right]\right] \Big\} \equiv 0$$

The last equality is obtained using (II.2.3), the discrete barotropic streamfunction time evolution equation. By the way, this shows that (II.2.3) is the only way do compute the streamfunction, otherwise the surface pressure forces will work. Nevertheless, since the elliptic equation verified by ψ is solved numerically by an iterative solver, the property is only satisfied at the precision required on the solver.

C.2 Conservation Properties on Ocean Thermodynamics

The numerical schemes are written such that the heat and salt contents are conserved by the internal dynamics (equations in flux form, second order centered finite differences). As a form flux is used to compute the temperature and salinity, the quadratic form of these quantities (i.e. their variance) is globally conserved, too. There is generally no strict conservation of mass, as the equation of state is non linear with respect to *T* and *S*. In practice, the mass is conserved with a very good accuracy.

C.2-a Tracer Conservation by Advective Term

$$\int_D \nabla.\left(T\,\mathbf{U}\right) dv$$

$$\equiv \sum_{i,j,k} \Big\{ \frac{1}{e_{1T} e_{2T} e_{3T}} \Big(\delta_i\left[e_{2u} e_{3u} \overline{T}^{i+1/2} u\right] + \delta_j\left[e_{1v} e_{3v} \overline{T}^{j+1/2} v\right] \Big) + \frac{1}{e_{3T}} \delta_k\left[\overline{T}^{k+1/2} w\right] \Big\} \; e_{1T} e_{2T} e_{3T}$$

$$\equiv \sum_{i,j,k} \Big\{ \delta_i\left[e_{2u} e_{3u} \overline{T}^{i+1/2} u\right] + \delta_j\left[e_{1v} e_{3v} \overline{T}^{j+1/2} v\right] + \delta_k\left[e_{1T} e_{2T} \overline{T}^{k+1/2} w\right] \Big\}$$

$$\equiv 0$$

C.2-b Variance of Tracer Conservation by Advective Term

$$\int_D T\,\nabla.\left(T\,\mathbf{U}\right) dv$$

$$\equiv \sum_{i,j,k} T \Big\{ \delta_i\left[e_{2u} e_{3u} \overline{T}^{i+1/2} u\right] + \delta_j\left[e_{1v} e_{3v} \overline{T}^{j+1/2} v\right] + \delta_k\left[e_{1T} e_{2T} \overline{T}^{k+1/2} w\right] \Big\}$$

$$\equiv \sum_{i,j,k} \Big\{ -e_{2u} e_{3u} \overline{T}^{i+1/2} u\, \delta_{i+1/2}[T] - e_{1v} e_{3v} \overline{T}^{j+1/2} v\, \delta_{j+1/2}[T] - e_{1T} e_{2T} \overline{T}^{k+1/2} w\, \delta_{k+1/2}[T] \Big\}$$

$$\equiv \sum_{i,j,k} -\frac{1}{2} \Big\{ e_{2u} e_{3u}\, u\, \delta_{i+1/2}\left[T^2\right] + e_{1v} e_{3v}\, v\, \delta_{j+1/2}\left[T^2\right] + e_{1T} e_{2T}\, w\, \delta_{k+1/2}\left[T^2\right] \Big\}$$

$$\equiv \sum_{i,j,k} \frac{1}{2} T^2 \Big\{ \delta_i\left[e_{2u} e_{3u} u\right] + \delta_j\left[e_{1v} e_{3v} v\right] + \delta_k\left[e_{1T} e_{2T} w\right] \Big\}$$

$$\equiv 0$$

C.3 Conservation Properties on Lateral Momentum Physics

The discrete formulation of the horizontal diffusion of momentum ensures the conservation of potential vorticity and horizontal divergence and the dissipation of the square of these quantities (i.e. enstrophy and the variance of the horizontal divergence) as well as the dissipation of the horizontal kinetic energy. In particular, when the eddy coefficients are horizontally uniform, it ensures a complete separation of vorticity and horizontal divergence fields, so that diffusion (dissipation) of vorticity (enstrophy) does not generate horizontal divergence (variance of the horizontal divergence) and *vice versa*.

These properties of the horizontal diffusive operator are a direct consequence of properties (II.1.8) and (II.1.9). When the vertical curl of the horizontal diffusion of momentum (discrete sense) is taken, the term associated to the horizontal gradient of the divergence is zero locally.

C.3-a Conservation of Potential Vorticity

The lateral momentum diffusion term conserves the potential vorticity :

$$\int_D \frac{1}{e_3}\mathbf{k}\cdot\nabla\times\left[\nabla_h\left(A^{lm}\,\chi\right)-\nabla_h\times\left(A^{lm}\,\zeta\,\mathbf{k}\right)\right]dv$$

$$=\int_D -\frac{1}{e_3}\mathbf{k}\cdot\nabla\times\left[\nabla_h\times\left(A^{lm}\,\zeta\,\mathbf{k}\right)\right]dv$$

$$\equiv\sum_{i,j}\left\{\,\delta_{i+1/2}\left[\frac{e_{2v}}{e_{1v}e_{3v}}\,\delta_i\left[A_f^{lm}e_{3f}\zeta\right]\right]+\delta_{j+1/2}\left[\frac{e_{1u}}{e_{2u}e_{3u}}\,\delta_j\left[A_f^{lm}e_{3f}\zeta\right]\right]\right\}$$

Using (II.1.10), it follows:

$$\equiv\sum_{i,j,k}-\left\{\,\frac{e_{2v}}{e_{1v}e_{3v}}\,\delta_i\left[A_f^{lm}e_{3f}\zeta\right]\delta_i[1]+\frac{e_{1u}}{e_{2u}e_{3u}}\,\delta_j\left[A_f^{lm}e_{3f}\zeta\right]\delta_j[1]\right\}\qquad\equiv 0$$

C.3-b Dissipation of Horizontal Kinetic Energy

The lateral momentum diffusion term dissipates the horizontal kinetic energy:

$$\int_D \mathbf{U}_h\cdot\left[\nabla_h\left(A^{lm}\chi\right)-\nabla_h\times\left(A^{lm}\,\zeta\mathbf{k}\right)\right]dv$$

$$\equiv\sum_{i,j,k}\left\{\frac{1}{e_{1u}}\delta_{i+1/2}\left[A_T^{lm}\chi\right]-\frac{1}{e_{2u}e_{3u}}\delta_j\left[A_f^{lm}e_{3f}\zeta\right]\right\}e_{1u}e_{2u}e_{3u}\,u+\left\{\frac{1}{e_{2u}}\delta_{j+1/2}\left[A_T^{lm}\chi\right]+\frac{1}{e_{1v}e_{3v}}\delta_i\left[A_f^{lm}e_{3f}\zeta\right]\right\}e_{1v}e_{2u}e_{3v}\,v$$

$$\equiv\sum_{i,j,k}\left\{\,e_{2u}e_{3u}\,u\,\delta_{i+1/2}\left[A_T^{lm}\chi\right]-e_{1u}\,u\,\delta_j\left[A_f^{lm}e_{3f}\zeta\right]\right\}+\left\{\,e_{1v}e_{3v}\,v\,\delta_{j+1/2}\left[A_T^{lm}\chi\right]+e_{2v}\,v\,\delta_i\left[A_f^{lm}e_{3f}\zeta\right]\right\}$$

$$\equiv\sum_{i,j,k}-\left(\,\delta_i\left[e_{2u}e_{3u}\,u\right]+\delta_j\left[e_{1v}e_{3v}\,v\right]\right)A_T^{lm}\chi-\left(\,\delta_{i+1/2}\left[e_{2v}\,v\right]-\delta_{j+1/2}\left[e_{1u}\,u\right]\right)A_f^{lm}e_{3f}\zeta$$

$$\equiv\sum_{i,j,k}-A_T^{lm}\,\chi^2\,e_{1T}e_{2T}e_{3T}-A_f^{lm}\,\zeta^2\,e_{1f}e_{2f}e_{3f}\qquad\leq 0$$

C.3-c Dissipation of Enstrophy

The lateral momentum diffusion term dissipates the enstrophy when the eddy coefficients are horizontally uniform:

$$\int_D \zeta\,\mathbf{k}\cdot\nabla\times\left[\nabla_h\left(A^{lm}\,\chi\right)-\nabla_h\times\left(A^{lm}\,\zeta\,\mathbf{k}\right)\right]dv$$

$$=A^{lm}\int_D \zeta\mathbf{k}\cdot\nabla\times\left[\nabla_h\times\left(\zeta\,\mathbf{k}\right)\right]dv$$

$$\equiv A^{lm}\sum_{i,j,k}\zeta\,e_{3f}\left\{\,\delta_{i+1/2}\left[\frac{e_{2v}}{e_{1v}e_{3v}}\,\delta_i\left[e_{3f}\zeta\right]\right]+\delta_{j+1/2}\left[\frac{e_{1u}}{e_{2u}e_{3u}}\,\delta_j\left[e_{3f}\zeta\right]\right]\right\}$$

Using (II.1.10), it becomes :

$$\equiv -A^{lm}\sum_{i,j,k}\left\{\,\left(\frac{1}{e_{1v}e_{3v}}\,\delta_i\left[e_{3f}\zeta\right]\right)^2 e_{1v}e_{2v}e_{3v}+\left(\frac{1}{e_{2u}e_{3u}}\,\delta_j\left[e_{3f}\zeta\right]\right)^2 e_{1u}e_{2u}e_{3u}\right\}\qquad\leq 0$$

C.3-d Conservation of Horizontal Divergence

When the horizontal divergence of the horizontal diffusion of momentum (discrete sense) is taken, the term associated to the vertical curl of the vorticity is zero locally, due to (II.1.8). The resulting term conserves the χ and dissipates χ^2 when the eddy coefficients are horizontally uniform.

$$\int_D \nabla_h\cdot\left[\nabla_h\left(A^{lm}\,\chi\right)-\nabla_h\times\left(A^{lm}\,\zeta\,\mathbf{k}\right)\right]dv=\int_D \nabla_h\cdot\nabla_h\left(A^{lm}\,\chi\right)dv$$

$$\equiv\sum_{i,j,k}\left\{\delta_i\left[A_u^{lm}\frac{e_{2u}e_{3u}}{e_{1u}}\delta_{i+1/2}[\chi]\right]+\delta_j\left[A_v^{lm}\frac{e_{1v}e_{3v}}{e_{2v}}\delta_{j+1/2}[\chi]\right]\right\}$$

Using (II.1.10), it follows:

$$\equiv\sum_{i,j,k}-\left\{\,\frac{e_{2u}e_{3u}}{e_{1u}}A_u^{lm}\delta_{i+1/2}[\chi]\;\delta_{i+1/2}[1]+\frac{e_{1v}e_{3v}}{e_{2v}}A_v^{lm}\delta_{j+1/2}[\chi]\;\delta_{j+1/2}[1]\right\}\qquad\equiv 0$$

C.3-e Dissipation of Horizontal Divergence Variance

$$\int_D \chi \nabla_h \cdot \left[\nabla_h \left(A^{lm} \chi \right) - \nabla_h \times \left(A^{lm} \zeta \mathbf{k} \right) \right] dv = A^{lm} \int_D \chi \nabla_h \cdot \nabla_h (\chi) dv$$

$$\equiv A^{lm} \sum_{i,j,k} \frac{1}{e_{1T} e_{2T} e_{3T}} \chi \left\{ \delta_i \left[\frac{e_{2u} e_{3u}}{e_{1u}} \delta_{i+1/2}[\chi] \right] + \delta_j \left[\frac{e_{1v} e_{3v}}{e_{2v}} \delta_{j+1/2}[\chi] \right] \right\} e_{1T} e_{2T} e_{3T}$$

Using (II.1.10), it turns out to be:

$$\equiv -A^{lm} \sum_{i,j,k} \left\{ \left(\frac{1}{e_{1u}} \delta_{i+1/2}[\chi] \right)^2 e_{1u} e_{2u} e_{3u} + \left(\frac{1}{e_{2v}} \delta_{j+1/2}[\chi] \right)^2 e_{1v} e_{2v} e_{3v} \right\} \leq 0$$

C.4 Conservation Properties on Vertical Momentum Physics

As for the lateral momentum physics, the continuous form of the vertical diffusion of momentum satisfies the several integral constraints. The first two are associated to the conservation of momentum and the dissipation of horizontal kinetic energy:

$$\int_D \frac{1}{e_3} \frac{\partial}{\partial k} \left(\frac{A^{vm}}{e_3} \frac{\partial \mathbf{U}_h}{\partial k} \right) dv = \vec{\mathbf{0}}$$

and

$$\int_D \mathbf{U}_h \cdot \frac{1}{e_3} \frac{\partial}{\partial k} \left(\frac{A^{vm}}{e_3} \frac{\partial \mathbf{U}_h}{\partial k} \right) dv \leq 0$$

The first property is obvious. The second results from:

$$\int_D \mathbf{U}_h \cdot \frac{1}{e_3} \frac{\partial}{\partial k} \left(\frac{A^{vm}}{e_3} \frac{\partial \mathbf{U}_h}{\partial k} \right) dv$$

$$\equiv \sum_{i,j,k} \left(u \delta_k \left[\frac{A_u^{vm}}{e_{3uw}} \delta_{k+1/2}[u] \right] e_{1u} e_{2u} + v \delta_k \left[\frac{A_v^{vm}}{e_{3vw}} \delta_{k+1/2}[v] \right] e_{1v} e_{2v} \right)$$

as the horizontal scale factor do not depend on k, it follows:

$$\equiv -\sum_{i,j,k} \left(\frac{A_u^{vm}}{e_{3uw}} \left(\delta_{k+1/2}[u] \right)^2 e_{1u} e_{2u} + \frac{A_v^{vm}}{e_{3vw}} \left(\delta_{k+1/2}[v] \right)^2 e_{1v} e_{2v} \right) \leq 0$$

The vorticity is also conserved. Indeed:

$$\int_D \frac{1}{e_3} \mathbf{k} \cdot \nabla \times \left(\frac{1}{e_3} \frac{\partial}{\partial k} \left(\frac{A^{vm}}{e_3} \frac{\partial \mathbf{U}_h}{\partial k} \right) \right) dv$$

$$\equiv \sum_{i,j,k} \frac{1}{e_{3f}} \frac{1}{e_{1f} e_{2f}} \left\{ \delta_{i+1/2} \left(\frac{e_{2v}}{e_{3v}} \delta_k \left[\frac{1}{e_{3vw}} \delta_{k+1/2}[v] \right] \right) - \delta_{j+1/2} \left(\frac{e_{1u}}{e_{3u}} \delta_k \left[\frac{1}{e_{3uw}} \delta_{k+1/2}[u] \right] \right) \right\} e_{1f} e_{2f} e_{3f} \equiv 0$$

If the vertical diffusion coefficient is uniform over the whole domain, the enstrophy is dissipated, i.e.

$$\int_D \zeta \mathbf{k} \cdot \nabla \times \left(\frac{1}{e_3} \frac{\partial}{\partial k} \left(\frac{A^{vm}}{e_3} \frac{\partial \mathbf{U}_h}{\partial k} \right) \right) dv = 0$$

This property is only satisfied in z-coordinates:

$$\int_D \zeta \mathbf{k} \cdot \nabla \times \left(\frac{1}{e_3} \frac{\partial}{\partial k} \left(\frac{A^{vm}}{e_3} \frac{\partial \mathbf{U}_h}{\partial k} \right) \right) dv$$

$$\equiv \sum_{i,j,k} \zeta \, e_{3f} \left\{ \delta_{i+1/2} \left(\frac{e_{2v}}{e_{3v}} \delta_k \left[\frac{A_v^{vm}}{e_{3vw}} \delta_{k+1/2}[v] \right] \right) - \delta_{j+1/2} \left(\frac{e_{1u}}{e_{3u}} \delta_k \left[\frac{A_u^{vm}}{e_{3uw}} \delta_{k+1/2}[u] \right] \right) \right\}$$

$$\equiv \sum_{i,j,k} \zeta \, e_{3f} \left\{ \frac{1}{e_{3v}} \delta_k \left[\frac{A_v^{vm}}{e_{3vw}} \delta_{k+1/2} \left[\delta_{i+1/2} [e_{2v} v] \right] \right] - \frac{1}{e_{3u}} \delta_k \left[\frac{A_u^{vm}}{e_{3uw}} \delta_{k+1/2} \left[\delta_{j+1/2} [e_{1u} u] \right] \right] \right\}$$

Using the fact that the vertical diffusive coefficients are uniform and that in z-coordinates, the vertical scale factors do not depends on i and j so that: $e_{3f} = e_{3u} = e_{3v} = e_{3T}$ and $e_{3w} = e_{3uw} = e_{3vw}$, it follows:

$$\equiv A^{vm} \sum_{i,j,k} \zeta \, \delta_k \left[\frac{1}{e_{3w}} \delta_{k+1/2} \left[\delta_{i+1/2} [e_{2v} v] - \delta_{j+1/2} [e_{1u} u] \right] \right]$$

$$\equiv -A^{vm} \sum_{i,j,k} \frac{1}{e_{3w}} \left(\delta_{k+1/2} [\, \zeta] \right)^2 e_{1f} e_{2f} \quad \leq 0$$

Similarly, the horizontal divergence is obviously conserved:

$$\int_D \nabla \cdot \left(\frac{1}{e_3} \frac{\partial}{\partial k} \left(\frac{A^{vm}}{e_3} \frac{\partial \mathbf{U}_h}{\partial k} \right) \right) dv = 0$$

and the square of the horizontal divergence decreases (i.e. the horizontal divergence is dissipated) if vertical diffusion coefficient is uniform over the whole domain:

$$\int_D \chi \, \nabla \cdot \left(\frac{1}{e_3} \frac{\partial}{\partial k} \left(\frac{A^{vm}}{e_3} \frac{\partial \mathbf{U}_h}{\partial k} \right) \right) dv = 0$$

This property is only satisfied in z-coordinates:

$$\int_D \chi \, \nabla \cdot \left(\frac{1}{e_3} \frac{\partial}{\partial k} \left(\frac{A^{vm}}{e_3} \frac{\partial \mathbf{U}_h}{\partial k} \right) \right) dv$$

$$\equiv \sum_{i,j,k} \frac{\chi}{e_{1T} e_{2T}} \left\{ \delta_{i+1/2} \left(\frac{e_{2u}}{e_{3u}} \delta_k \left[\frac{A_u^{vm}}{e_{3uw}} \delta_{k+1/2} [u] \right] \right) + \delta_{j+1/2} \left(\frac{e_{1v}}{e_{3v}} \delta_k \left[\frac{A_v^{vm}}{e_{3vw}} \delta_{k+1/2} [v] \right] \right) \right\} e_{1T} e_{2T} e_{3T}$$

$$\equiv A^{vm} \sum_{i,j,k} \chi \left\{ \delta_{i+1/2} \left(\delta_k \left[\frac{1}{e_{3uw}} \delta_{k+1/2} [e_{2u} u] \right] \right) + \delta_{j+1/2} \left(\delta_k \left[\frac{1}{e_{3vw}} \delta_{k+1/2} [e_{1v} v] \right] \right) \right\}$$

$$\equiv -A^{vm} \sum_{i,j,k} \frac{\delta_{k+1/2} [\chi]}{e_{3w}} \left\{ \delta_{k+1/2} [\, \delta_{i+1/2} [e_{2u} u] + \delta_{j+1/2} [e_{1v} v]] \right\}$$

$$\equiv -A^{vm} \sum_{i,j,k} \frac{1}{e_{3w}} \delta_{k+1/2} [\chi] \;\; \delta_{k+1/2} [\, e_{1T} e_{2T} \, \chi \,]$$

$$\equiv -A^{vm} \sum_{i,j,k} \frac{e_{1T} e_{2T}}{e_{3w}} \left(\delta_{k+1/2} [\chi] \right)^2 \quad \equiv 0$$

C.5 Conservation Properties on Tracer Physics

The numerical schemes used for tracer subgridscale physics are written such that the heat and salt contents are conserved (equations in flux form, second order centered finite differences). As a form flux is used to compute the temperature and salinity, the quadratic form of these quantities (i.e. their variance) globally tends to diminish. As for the advection term, there is generally no strict conservation of mass even if, in practice, the mass is conserved with a very good accuracy.

C.5-a Conservation of Tracers

constraint of conservation of tracers:

$$\int_D T \, \nabla . (A \, \nabla T) \, dv$$

$$\equiv \sum_{i,j,k} \left\{ \delta_i \left[A_u^{lT} \frac{e_{2u} e_{3u}}{e_{1u}} \delta_{i+1/2} [T] \right] + \delta_j \left[A_v^{lT} \frac{e_{1v} e_{3v}}{e_{2v}} \delta_{j+1/2} [T] \right] + \delta_k \left[A_w^{vT} \frac{e_{1T} e_{2T}}{e_{3T}} \delta_{k+1/2} [T] \right] \right\}$$

$$\equiv 0$$

C.5-b Dissipation of Tracer Variance

constraint of dissipation of tracer variance:

$$\int_D T \, \nabla . (A \, \nabla T) \, dv$$

$$\equiv \sum_{i,j,k} T \left\{ \delta_i \left[A_u^{lT} \frac{e_{2u} e_{3u}}{e_{1u}} \delta_{i+1/2} [T] \right] + \delta_j \left[A_v^{lT} \frac{e_{1v} e_{3v}}{e_{2v}} \delta_{j+1/2} [T] \right] + \delta_k \left[A_w^{vT} \frac{e_{1T} e_{2T}}{e_{3T}} \delta_{k+1/2} [T] \right] \right\}$$

$$\equiv -\sum_{i,j,k} \left\{ A_u^{lT} \left(\frac{1}{e_{1u}} \delta_{i+1/2} [T] \right)^2 e_{1u} e_{2u} e_{3u} + A_v^{lT} \left(\frac{1}{e_{2v}} \delta_{j+1/2} [T] \right)^2 e_{1v} e_{2v} e_{3v} + A_w^{vT} \left(\frac{1}{e_{3w}} \delta_{k+1/2} [T] \right)^2 e_{1w} e_{2w} e_{3w} \right\} \quad \leq 0$$

APPENDIX D CODING RULES

The "model life" is about ten years and its software, composed by about one hundred programs, is used by many people who are scientists or students and do not necessary know very well all computer aspects. Moreover, a well thought-out programme is easy to read and understand, less difficult to modify, produces fewer bugs and is easier to maintain. Therefore, it is essential that the model development follows some rules :

- well planned and designed
- well written
- well documented (both on- and off-line)
- maintainable
- easily portable
- flexible.

To satisfy part of these aims, OPA is written with a coding standard which is close to the ECMWF rule, named DOCTOR [Gibson,1986]. These rules present some advantages like :

- to provide a well presented program
- to enable the extraction of several levels of on-line documentation
- to use rules for variable names which allow recognition of their type (integer, real, parameter, common variables, etc.) so that debugging is facilitated.

The program structure

Each program begins with a set of headline comments containing :

- the program title
- the purpose of the routine
- the method and algorithms used
- the detail of input and output interfaces
- the external routines and functions used (if exist)
- references (if exist)
- the author name (s), the date of creation and of updates.
- Each program is splitted into several well separated sections and sub-sections with a underlined title and specific labelled statements.
- A program has not more than 200 to 300 lines.

Coding conventions

- Use of the universal language FORTRAN 5 ANSI 77, with non standard extensions, NAMELIST and FORTRAN 90 (matrix resolution algorithm), and some well-identified particular statements or functions for weak and massive parallelism, and vectorization
- A comment line begins with a uppercase character C at the first column. A space line must have a C. For the on-line documentation, comments are classified into three levels :

Overview, triggered by CCC in columns 1 to 3. Only the title and the purpose of the program are identified like that. This overview documentation can be extracted by the UNIX function : grep -e '^CCC' *

External, triggered by CC in columns 1 to 2, and which correspond to headlines of each programme, extracted by : grep -e '^CC' *

Internal which are all the comments, extracted by : grep -e '^C' *

- Statements GO TO, EQUIVALENCE are forbidden.
- A section is numbered with labels which are in agreement with the paragraph label and increase from the begin to the end of routine. Labels of a hundred (200, 201.. 220..) are reserved to a unique section. The FORTRAN 90 extension syntax DO/ENDDO is used except for multitasked do-loop. In this case labels 1000, 2000, ... are used. The FORMAT statement are labelled with numbers in the range 9000 to 9999.
- A continuation line begins with the character $ in column 6.
- All statements begin in column 7 with the following gaps :

2 spaces toward the right in a DO loop.

4 spaces toward the right in IF, ELSEIF, ELSE and ENDIF statements, with only 2 spaces for ELSE and ELSEIF lines. All IF statement must be followed by a ELSE statement.

Some spaces in the continuation line for alignment.

- Use of different labels for each DO loop statement.
- STOP must be well documented with the name of the subroutine or a number.

Naming Conventions.

The purpose of the naming conventions is to use prefix letters to classify model variables. These conventions allow to know easily the variable type and to identify them rapidly:

type / status	integer	real	logical	character	double precision	complex
global or common	**m n** *but not* **nam**	**a b e f g h o q** *to* **x** but not **s f**	**l** *but not* **lp ld ll**	**c** *but not* **cp cd cl com cim**	**d** *but not* **dp dd dl**	**y** *but not* **yp yd yl**
dummy argument	**k** *but not* **k f**	**p** *but not* **pp pf**	**ld**	**cd**	**dd**	**yd**
local variable	**i**	**z**	**l l**	**c l**	**cd**	**y l**
loop control	**j** *but not* **jp**					
parameter	**jp**	**pp**	**l p**	**c p**	**dp**	**y p**
statement function	**k f**	**s f**				

References

Gibson, J. K., 1986: Standard software development and maintenance. *Tech. memorandum*, Operational Dep., ECMWF, Reading, UK.

INDEX: CPP VARIABLES

INDEX: NAMELIST PARAMETERS