Planetary-Geostrophic Ocean Model: A User's Guide

R. M. Samelson
Woods Hole Oceanographic Institution, Woods Hole, MA 02543
rsamelson@whoi.edu
Geoffrey K. Vallis
University of California, Santa Cruz, CA 95064
vallis@cascade.ucsc.edu

November, 1995.

Abstract

This document describes a three-dimensional planetary geostrophic ocean model (PGOM). The model implements a particular form of the planetary-geostrophic equations which is well suited to fast numerical solution in a closed domain. Wind and thermodynamic forcing are allowed. The model can be run in either Cartesian or spherical geometry, and may cross the equator. This document briefly describes the model equations, their finite difference form, and the associated numerical code.

1 Equations of Motion

This section summarizes the equations implemented in the code, in Cartesian form, as described by Samelson and Vallis (1995a, 1995b). The spherical co-ordinate form, which is implemented in the code via a run-time switch, may be obtained from these by inserting the appropriate metric factors, nondimensionalizing distance by the Earth's radius, and treating x and y as angular longitude and latitude, respectively. The equations of motion are:

$$-fv = -p_x - \varepsilon u \tag{1.1}$$

$$fu = -p_y - \varepsilon v \tag{1.2}$$

$$0 = -p_z - \rho \tag{1.3}$$

$$u_x + v_y + w_z = 0 \tag{1.4}$$

$$T_t + uT_x + vT_y + wT_z = \kappa_v T_{zz} + \kappa_h \Delta_h T - \lambda \Delta_h^2 T \tag{1.5}$$

$$S_t + uS_x + vS_y + wS_z = \kappa'_v S_{zz} + \kappa'_h \Delta_h S - \lambda' \Delta_h^2 S \tag{1.6}$$

$$\rho = S - T \tag{1.7}$$

where

$$\Delta_h = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}. \tag{1.8}$$

Here (u, v, w) are the (x, y, z)-components of velocity, t is time, p is pressure divided by a constant reference density ρ_0, T is temperature, S is salinity, ρ is density, and subscripts x, y, z, and t denote partial derivatives. The parameters ε, κ_v, κ_h, λ, κ_v', κ_h', λ' are frictional and diffusive parameters. The frictional parameter ε may vary in x and y, although in the discussion below it is taken to be uniform. These equations are supplemented by a convective adjustment scheme, which removes static instabilities at each time step by vertically mixing grid-point fluid volumes to restore neutral stability.

The frictional-geostrophic relations (1.1) and (1.2) may be inverted for the horizontal velocities, giving

$$u = -\gamma(\varepsilon p_x + f p_y), \quad v = \gamma(f p_x - \varepsilon p_y) \tag{1.9}$$

where $\gamma = (f^2 + \varepsilon^2)^{-1}$.

Substituting these expressions into the continuity equation (1.4), gives

$$\mathcal{H}(p) = w_z \tag{1.10}$$

where

$$\mathcal{H}(p) = \varepsilon\gamma\Delta_h p + (f^2 - \varepsilon^2)\gamma^2\beta p_x - 2\varepsilon f\gamma^2\beta p_y \tag{1.11}$$

in the case of constant ε. A diagnostic equation for the barotropic pressure, P, where

$$P = \int_0^1 p\, dz \tag{1.12}$$

may then be obtained by integrating (1.10) vertically, yielding

$$\mathcal{H}(P) = W_1 - W_0 \tag{1.13}$$

where $W_1 = w_E$ and $W_0 = 0$ are the imposed vertical velocities at the top ($z = 1$) and bottom ($z = 0$) boundaries of the interior domain.

The equations are solved as follows. The elliptic equation (1.13) is solved for P by SOR, subject to the vertically-integrated no-normal-flow condition

$$\varepsilon P_n + f P_s = 0 \tag{1.14}$$

(where the subscripts n and s denote the outward normal and right-handed tangential derivatives, respectively) at the lateral boundaries. If the imposed Ekman pumping w_E is independent of time the elliptic equation (1.13) need be solved only once, at the beginning of the time-stepping integration. If the temperature T and salinity S is known at a given time, the density is computed from an equation of state, and the baroclinic pressure $p' \equiv p - P$ may be obtained by integrating the hydrostatic relation (1.3). The velocities are then calculated from the pressure $p = p' + P$ using (1.9) and (1.4). The no-heat-flux condition at lateral boundaries is a boundary condition on temperature,

$$-\kappa_h T_n + \lambda\Delta_h T_n = 0. \tag{1.15}$$

The vertical derivative of (1.9) yields the frictional-geostrophic analog of the thermal wind relation,

$$u_z = -\gamma(\varepsilon T_x + f T_y), \quad v_z = \gamma(f T_x - \varepsilon T_y) \tag{1.16}$$

and, consequently, the no-normal-flow boundary condition implies a second boundary condition on temperature,

$$\varepsilon T_n + f T_s = 0. \tag{1.17}$$

The presence of the biharmonic diffusion term in (1.5) allows the two conditions (1.15) and (1.17) to be simultaneously satisfied at each lateral boundary. When salinity S is included, (1.15) and (1.17) are also applied to (1.6), with T replaced by S.

1.1 Surface boundary layer and air-sea fluxes

The upper boundary conditions on temperature and normal (vertical) velocity for the interior equations are obtained from a simple 'slab' model of a frictional surface boundary layer. The boundary layer is taken to have fixed depth δ_E, with its base at the top of the interior domain, so the sea surface is at $z = 1 + \delta_E$. The vertically integrated Ekman balance is assumed to hold, so that the horizontal Ekman velocities are

$$(u_E, v_E) = (\tau^y, -\tau^x)/f\delta_E \tag{1.18}$$

where $\tau = (\tau^x, \tau^y)$ is the wind stress at the sea surface. The vertical velocity at the base of the boundary layer may be computed from the continuity relation,

$$w_E = \delta_E(u_{Ex} + v_{Ey}). \tag{1.19}$$

The default wind stress is zonal ($\tau^y = 0$), and has the form

$$\tau^x = \tau_0(f(y)/f_0)\sin 2\pi y, \tag{1.20}$$

so the Ekman flow is meridional ($u_E = 0$),

$$v_E = -(\tau_0/f_0\delta_E)\sin 2\pi y, \tag{1.21}$$

and the Ekman pumping has the form

$$w_E = w_{E0}\cos 2\pi y \tag{1.22}$$

where $w_{E0} = -2\pi\tau_0/f_0$.
The surface boundary layer temperature T_E is obtained from a vertically integrated thermodynamic equation,

$$T_{Et} + (u_A T_E)_x + (v_A T_E)_y = (F_T - F_i)/\delta_E \tag{1.23}$$

where F_T is the air-sea flux and F_i, the upward flux through the base of the boundary layer, is

$$F_i = (w_E T - \kappa_v T_z)_{z=1} \tag{1.24}$$

In order to allow the northward advection of warm surface boundary layer water by the western boundary current, the interior velocities at the base of the boundary layer are added to the Ekman flow to obtain the advection velocities in(1.23),

$$(u_A, v_A) = (u_E, v_E) + (u, v)_{z=1} \tag{1.25}$$

Since the horizontal advection terms in (1.23) are in flux form, the area integral of the horizontal advective terms vanishes, that is, the net change in heat content is due to fluxes across the upper and lower boundaries of the surface layer. However, the advection field is in effect slightly compressible, since the vertical motion driven by horizontal convergence of the interior flow is neglected in (1.22).

The air-sea flux F_T is obtained from T_E and an imposed atmospheric surface temperature T_s using the flux law:

$$F_T = -\gamma_{Ts}(T_E - T_s) \tag{1.26}$$

The air temperature defaults to a (piecewise-continous) linear function of y.

The freshwater flux w_0 (Huang, 1993) may also be specified at the sea-surface. The model salinity S is the difference from a given reference value S_0, so the zero-flux condition on salinity at the sea surface results in the freshwater flux boundary condition $F_S = w_0 S_0$ for precipitation ($w_0 < 0$) and evaporation ($w_0 > 0$). Precipitation has the local imposed air temperature T_s, while evaporation removes fluid with the local ocean surface temperature T_E.

1.2 Nondimensionalization

The equations and code should be regarded as nondimensional. The conversion to dimensional values requires scales for depth, length, Coriolis parameter, density, gravity, and vertical velocity. For example (respectively): $D = 5 \times 10^5$ cm, $L = 5 \times 10^8$ cm, $f_* = f(35^o\text{N}) = 8.4 \times 10^{-5}$ s^{-1}, $\rho_0 = 1$ g cm^{-3}, $g = 980$ cm s^{-2}, $W = 10^{-4}$ cm s^{-1}.

From these, the following dimensional scales may be derived for horizontal velocity, time, density variations, temperature variations, buoyancy frequency, heat flux and heat transport, respectively: $U = WL/D = 0.1$ cm s^{-1}, $t_* = D/W = L/U = 5 \times 10^9$ s $= 160$ yr, $\Delta\rho = (\rho_0 f_* UL)/(gD) = 8.6 \times 10^{-6}$ g cm^{-3} ~ 0.01 σ_θ, $\Delta T = \Delta\rho/(\rho_0\alpha_T) = 0.086$ K, $N_0 = (g\Delta\rho/(\rho_0 D))^{1/2} = 1.3 \times 10^{-4}$ s^{-1} $= 0.075$ cph, $H_f = 0.34$ W m^{-2}, $H_F = H_f L^2 = 0.8 \times 10^{13}$ W, where the values $\alpha_T = 10^{-4}$ K^{-1} and $c_p = 4000$ J kg^{-1} K^{-1} have been used for the thermal expansibility and specific heat of sea water.

Dimensional scales for the vertical, horizontal, and biharmonic horizontal thermal diffusivities are then $K_v = WD = 50$ cm^2 s^{-1}, $K_h = UL = 5 \times 10^7$ cm^2 s^{-1}, $\Lambda = UL^3 = 1.25 \times 10^{25}$ cm^4 s^{-1}. The friction parameter ε and the air-sea flux coefficient γ_{Ts} are nondimensionalized by f and W, respectively.

2 Finite Difference Equations

Advection is effected using centered differencing in flux form, except at the upper boundary where an upwind scheme is implemented. Time-stepping uses a second order Runge-Kutta algorithm. In the discussion following, if salinity is not explicitly mentioned its treatment is the same as that of temperature.

Let the domain be covered by a three-dimensional grid, $\{i, j, k\}$ denoting increments in the x-, y-, and z-directions respectively, and let temperature, T, salinity S, density, ρ, and pressure, p, be defined on that grid. Let Δx, Δy and Δz be the respective grid increments. If i_{max} and j_{max} are the number of grid points in the x and y directions, the lateral boundaries are considered to lie at $\{i = 1 + 1/2, i_{max} - 1/2\}$ and $\{j = 1 + 1/2, j_{max} - 1/2\}$, namely one half a grid interval in from the edge.

Define the following variables:

$$\hat{T}_{i-\frac{1}{2},j,k} = \frac{1}{2}(T_{i,j,k} + T_{i-1,j,k}) \tag{2.1}$$

$$\tilde{T}_{i,j-\frac{1}{2},k} = \frac{1}{2}(T_{i,j,k} + T_{i,j-1,k}) \tag{2.2}$$

$$\overline{T}_{i,j,k-\frac{1}{2}} = \frac{1}{2}(T_{i,j,k} + T_{i,j,k-1}) \tag{2.3}$$

and similarly for p and ρ.

Define also the difference operators:

$$\delta_x \phi_{i,j,k} = (\phi_{i+\frac{1}{2},j,k} - \phi_{i-\frac{1}{2},j,k})/\Delta x, \tag{2.4}$$

$$\delta_y \phi_{i,j,k} = (\phi_{i,j+\frac{1}{2},k} - \phi_{i,j-\frac{1}{2},k})/\Delta y, \tag{2.5}$$

where $\phi_{i,j,k}$ is an arbitrary field.

Then the horizontal velocities are defined on a staggered grid, obtained from pressure on as follows:

$$u_{i-\frac{1}{2},j,k} = -\gamma_j \left(\varepsilon(p_{i,j,k} - p_{i-1,j,k})/\Delta x + f_j(\hat{p}_{i-\frac{1}{2},j+1,k} - \hat{p}_{i-\frac{1}{2},j-1,k})/2\Delta y\right) \tag{2.6}$$

and

$$v_{i,j-\frac{1}{2},k} = \gamma_{j-\frac{1}{2}} \left(f_{j-\frac{1}{2},k}(\tilde{p}_{i+1,j-\frac{1}{2},k} - \tilde{p}_{i-1,j-\frac{1}{2},k})/2\Delta x - \varepsilon(p_{i,j,k} - p_{i,j-1,k})/\Delta y\right) \tag{2.7}$$

where $\gamma_j = f_j^2 + \varepsilon^2$, with f_j the Coriolis parameter.

The vertical velocity is staggered between the horizontal velocity levels. It is obtained by integrating the mass conservation equation,

$$(w_{i,j,k+\frac{1}{2}} - w_{i,j,k-\frac{1}{2}})/\Delta z = \delta_x u_{i,j,k} + \delta_y v_{i,j,k}, \tag{2.8}$$

with the vertical velocity set equal to zero at the lower boundary. The vertical grid need not be uniform, and a stretched grid is in fact normally used to give enhanced resolution in the thermocline. The vertical grid interval Δz is then in general a function of the index k.

The finite difference form of the barotropic elliptic equation is obtained by substituting (2.6) and (2.7) into the mass conservation equation, and summing over vertical levels. The form of resulting difference equation is not particularly informative. The boundary conditions on this elliptic equation are no-normal flow, to be applied on the boundary columns and rows where u and v are repectively defined, namely $i = 1 + 1/2$ and $i = i_{max} - 1/2$, and $j = 1 + 1/2$ and $j = j_{max} - 1/2$. The difference form leads to a cyclic tridiagonal problem for the boundary values of p, with the interior values of the pressure appearing as 'forcing' terms on the right-hand-side, which can be rapidly solved. The elliptic problem is solved by successive over relaxation (SOR); multigrid methods would also be appropriate if speed were a consideration. The correct boundary values are obtained by recalculating them after each internal SOR iteration until the solution converges.
The (nondimensional) hydrostatic equation is differenced as follows:

$$(p_{i,j,k+1} - p_{i,j,k})/\Delta z = \overline{\rho}_{i,j,k+\frac{1}{2}} \tag{2.9}$$

The advection of temperature or salinity uses centered differences in flux form, ensuring conservation of the L_1 norm of the advected quantity. The finite difference form of the advection term is:

$$(u_{i+\frac{1}{2},j,k}\hat{T}_{i+\frac{1}{2},j,k} - u_{i-\frac{1}{2},j,k}\hat{T}_{i-\frac{1}{2},j,k})/\Delta x \tag{2.10}$$

$$+ (v_{i,j+\frac{1}{2},k}\tilde{T}_{i,j+\frac{1}{2},k} - v_{i,j-\frac{1}{2},k}\tilde{T}_{i,j-\frac{1}{2},k})/\Delta y \tag{2.11}$$

$$+ (w_{i,j,k+\frac{1}{2}}\overline{T}_{i,j,k+\frac{1}{2}} - w_{i,j,k-\frac{1}{2}}\overline{T}_{i,j,k-\frac{1}{2}})/\Delta z \tag{2.12}$$

The harmonic horizontal diffusion terms are obtained using a standard five-point Laplacian stencil, and the biharmonic terms are obtained by iterating this once. The boundary conditions of no-normal flow and no normal diffusive flux at the boundary determine the boundary values, the former requiring the application of the cyclic tridiagonal solver. Vertical diffusion is also implemented in a standard way. Normally a no-heat-flux condition is applied at the bottom of the domain, whereas the temperature at the top is determined from the Ekman layer dynamics.
The overall time-stepping procedure is as follows. Given a wind field, the barotropic pressure field is obtained by solving the difference form of the barotropic elliptic equation. Given also an initial temperature field (and salinity field if appropriate) the density field is diagnosed from an equation of state, and the complete pressure field is then obtained by vertically integrating the hydrostatic equation (2.9). The horizontal and vertical velocities are then obtained from near-geostrophic balance, (2.6, 2.7) and mass conservation (2.8). These velocities are used to advance the temperature (and salinity) fields one timestep, and the process is repeated.

3 Model Basics

3.1 Files and overall structure

This section gives a brief outline of the structure of the model.

The code is collected into five groups of subroutines (for convenience only):

1. **pgmain.f:** The main calling program, plus most of the integration routines
2. **pguvwp.f:** Contains routines that evaluate the velocity from the temperature and salinity fields. This includes the elliptic solver (for the barotropic pressure) and the routines which use hydrostatic and near-geostrophic balance to obtain the velocity.
3. **pginit.f:** Contains the initialization routines, including the routine that sets up the vertical grid and various other unchaninging arrays.
4. **pgio_cdf.f, pgio_bin.f:** These contain I/O routines in NetCDF and unformatted binary formats, respectively. Only one of these files should be linked during compilation as they contain subroutines with identical names. Note that the code can be used to convert file formats by compiling with input and output routines in different formats.
5. **pgother.f:** Contains miscellaneous other routines, including interpolation routines to double the resolution.

In addition the code contains:

1. **const.name:** A namelist containing most of the important parameters used in the model.
2. **params.inc:** An include file that determines the model resolution during compilation.
3. **common.inc:** This contains a couple of common blocks which contain static parameters and arrays that do not change through the course of a model integration. (The common blocks are replaced with modules in the f90 version.)
4. **Makefile:** A simple makefile for UNIX users.

In addition to comments in the source files, the documentation consists of:

1. **pgman.tex, pgman.ps:** LaTex and PostScript versions of this guide.
2. **Sphere.log, Cartes.log:** Some sample log files with model output for reference and comparison. The const.name files used in these examples are also included.

The code flows roughly as follows:

pgmain: main calling program
initlz: Set constants and initialize arrays
pgin: Read initial data, if any
init1: Set forcing and initial values
ell2d: Solve elliptic problem for barotropic velocity

Begin main timestepping loop:

pgstep: Do one t-s step (split: 2nd order Runge Kutta + 1 Euler)
pguvw: Get velocities
convct: Convective adjustment
boundb: Lateral boundary condition (no-normal-flow) solver
pgadv: Density and salinity advection
ekman: Ekman advection and surface flux
eustpr: Euler forward step (repeat for R-K step)
fluxbc: Bottom boundary condition
pgvdf: Vertical diffusion
boundb: Lateral boundary condition (no-normal-flow) solver
pghdf: Horizontal diffusion
eustpr: Euler forward step (once for Euler forward)
End timestepping loop

pgout: Write tc arrays

3.2 Dimensions: params.inc

NX, NY, and NZ are respectively the number of zonal, meridional, and vertical ρ points. There are (NX-2)*(NY-2)*NZ interior grid cells, since the lateral rigid boundaries are halfway between the outermost ρ points. The surface boundary layer is vertical point NZ+1, and the bottom boundary is vertical point 0. Thus the resolution 20x20x14 in the example below will give a horizontal grid resolution of 3 degrees, when ispher=1 and alx=axly=54 in the const.name file, and 14 vertical interior levels plus the surface boundary layer at NZ+1=15.
Example params.inc

```
!  Grid dimensions:  NX,NY,NZ - east, north, up
      parameter (NX=20,NY=20,NZ=14)
      parameter (KTOT=2*(NX+NY-2))
```

3.3 Namelist: const.name

Most of the main user-changeable parameter are set in the namelist, *const.name.* This contains the following parameters.

ifile, ofile: Names of input and output files. These must have exactly six characters. For example: ifile='input1', ofile='outpt0'. The corresponding files will have the default extensions '.cdf' for NetCDF format and '.bin' for binary format. For example: 'input1.cdf' and 'outpt0.cdf'.

isal: salinity switch. If isal=0, no salinity; if isal=1, salinity is present.

iconv: Convective adjustment switch. iconv=1 (usual value) if convective adjustment is turned on, iconv = 0 turns off convective adjustment.

iekman: Determines how the surface layer advection velocity is calculated. See subroutines init1 and subroutine velo. Typical value: 1.

ispher: If ispher = 1, spherical model. If ispher=0, cartesian model.

ipr: If ipr=0 (usual value) little printing is done. For other values, some diagnostic printing takes place.

nt: Number of timesteps.

t0: Initial time. Useful when continuing runs, or for time-dependent runs.

dt: Timestep.

alx,aly: The domain size in the x- and y-directions. In the Cartesian model it is non-dimensional length units, and in the spherical model in degrees. Typical values: Cartesian model: alx=1, aly=1. Spherical model: alx=54, aly=54.

adz: Stretching parameter for vertical grid. See subroutine gridz. If adz=100 or adz=101, special vertical grids are put in 'by hand.' Otherwise, a simple analytic form is put in that enhances resolution in the upper ocean. Typical value, adz=3.

ys: Location of southern boundary of ocean. Same units as alx and aly for either spherical or Cartesian.

f0,beta: Values of f and beta in Cartesian model. Ignored if ispher=1.

akvt, akht, alht: Vertical, horizontal (harmonic) and horizontal (bi-harmonic) diffusion coefficients of temperature.

akvs, akhs, alhs: Vertical, horizontal (harmonic) and horizontal (bi-harmonic) diffusion coefficients of salinity.

gammats, gammatb: Surface and bottom coefficients for temperature flux relaxation schemes.

gammass,gammasb: As for gammats, gammatb but for salinity.

ts1,ts2: These parameters determine the 'atmospheric' value of temperature toward which the surface value relaxes. See subroutine init1.

tb0: This is the initial temperature value if the model is started from scratch, also bottom value for flux condition.

ss0: This determines the value of salinity field toward which the surface value relaxes.

sb0: This is the initial salinity value if the model is started from scratch, also bottom value for flux condition.

we0: The amplitude of the wind forcing.

alwe: Determines the horizontal structure of the wind forcing. Currently, it set the meridional wavenumber in a simple analytic function. See subroutine init1.

de: The depth of the surface boundary (Ekman) layer.

w00: The amplitude of the freshwater forcing. See subroutine init1. Normally set equal to zero if isal=0.

scap0: Reference value of salinity. Typical value 2800.

epsi, epsb, depsb, epse, depse: Parameters determining the value of the linear Rayleigh drag coefficient ε in the momentum equation. The user should examine the source code before changing these parameters. They have the following meanings:

epsi: Interior value, the value used over most of the domain.

epsb: Value at lateral boundary.

depsb: Thickness of transition zone from interior to lateral boundary.

epse: Value at equator.

depse: Thickness of transition zone from extra-equatorial interior to equator.

omega1,tol1,maxits: The overrelaxation parameter used in the SOR scheme, the tolerance to which the solution is computed, and the maximum number of iterations.

Example Spherical const.name

```
$constn  ifile='spher1',ofile='spher2',
         isal=0,iconv=1,iekman=1,ispher=1,ipr=0,
         nt=10,t0=0.002,dt=2.e-4,
         alx=54.,aly=54.,adz=3.,ys=10.,f0=1.,beta=1.1,
         akvt=0.018,akht=0.65,alht=+1.e-5,
         akvs=0.001,akhs=0.001,alhs=+1.e-5,
         gammats=  5.6,gammatb= 0.,gammass=   0.,gammasb=   0.,
         ts1=-227.,ts2=-227.,tb0=-200.,ss0=   0.,sb0=  0.,
         we0=  3.,alwe=2.,de=   0.0053,w00=   0.,scap0=  2800.,
         epsi=0.08,epsb=0.08,depsb=0.1,epse=0.,depse=0.1,
         omega1=1.2,tol1=1.e-11,maxits=4999
$end
```

Example Cartesian const.name

```
 $constn  ifile=' ',ofile='carte1',
          isal=0,iconv=1,iekman=1,ispher=0,ipr=0,
          nt=10,t0=0.,dt=5.e-5,
          alx=1.,aly=1.,adz=3.,ys=0.,f0=1.,beta=1.1,
          akvt=0.005,akht=0.040,alht=+1.e-5,
          akvs=0.001,akhs=0.001,alhs=+1.e-5,
          gammats=  5.,gammatb= 0.,gammass=   0.,gammasb=   0.,
          ts1=-200.,ts2=-200.,tb0=-180.,ss0=   0.,sb0=  0.,
          we0=  1.,alwe=2.,de=   0.005,w00=    0.,scap0=  2800.,
          epsi=0.05,epsb=0.05,depsb=0.1,epse=0.,depse=0.1,
          omega1=1.2,tol1=1.e-11,maxits=4999
 $end
```

3.4 Running the Model: Examples

- *To run the model from default initial state:*

  Set resolution by setting the parameters in file params.inc.

  Compile (using the UNIX 'make' command and the file Makefile).

  Set parameters in the file const.name. This includes choice of spherical / cartesian model, number of timesteps and so on. Set the ifile parameter to a single blank character to start from the default (uniform) initial state.

  Execute pgom.x

- *To continue a run with the same parameters:*

  Set the ifile parameter in const.name to read in the previously outputted data file. Set intial time t0 appropriately in const.name. Execute pgom.x

- *To run a pre-existing executable at the same resolution with different parameters:*

  Change the appropriate parameters in the namelist const.name. Execute pgom.x

- *To change model resolution:*

  Change the included file params.inc. Recompile the entire code. Make sure that every file is recompiled. (This can be ensured by typing 'touch *.f' before executing the makefile.) Recompilation is not necessary for the f90 version, since f90 arrays are dynamically allocated.

- *To change the wind field:*

  Edit the subroutine init1b.f. Recompile the code. Currently, the wind field is assumed to be temporally unchanging. To change this would require that the elliptic solver

for the barotropic streamfunction be called repeatedly during the model integration (unless only the wind amplitude varied, which would significantly reduce the additional computational load and the amount of recoding required).

- *To change the thermodynamic forcing (surface fluxes)*:

  Edit the subroutine init1b.f. Recompile the code. Currently, the surface temperature and salinity field are assumed temporally unchanging. This assumption could be relaxed with minor recoding. Note that some aspects of the surface fluxes (e.g. the values of ts1 etc) can be changed simply by editing the namelist.

4 Final Comments

Finally, we caution the user that the model should not be used as a 'black-box,' and some familiarity with both the code and the underlying physics and dynamics is expected. For example, care should be taken in regard to the following processes: (i) The biharmonic diffusion. This does not necessarily mix downgradient, and if its value is too large un-mixing could occur; (ii) The effects of the 'compressible' Ekman layer; (iii) The Ekman pumping should integrate to zero, else global mass conservation is lost and the elliptic solver may not find a solution. There are other similar pitfalls for the unwary. And please notify the authors of any bugs or errors in the code or the documentation.

References

Huang, R. X., 1993. Real freshwater flux as the upper boundary condition for the salinity balance and thermohaline circulation forced by evaporation and precipitation. *J. Phys. Oceanogr.*, **23**, 2428-2446.

Samelson, R., and G. K. Vallis, 1995. Planetary geostrophic equations for large-scale ocean modelling. *J. Phys. Oceanogr.*, submitted.

Samelson, R., and G. K. Vallis, 1995. Large-scale circulation with small diapycnal diffusion: the two-thermocline limit. *J. Mar. Res.*, submitted.