User Manual for PSOM

$\sim$ et. al

2017

Contents

1 Introduction

2 Headstart

3 Configure compile run sequence

4 Test Cases

4.1 Wiggles

4.2 NA

4.3 Shelfbreak

5 Setting up your PSOM simulation

5.1 Create your experiment directory

For every experiment you want to conduct, create a directory that will contain the source files specific to this experiment. As a example, you want to create an experiment named "my_experiment". First, create a directory `code/my_experiment`, in which you will create two subdirectories `/inc` and `/src` . You can either create this directory manually, or, in `code/`, you can run:

```
# Copies the template directory
cp -r expe_template my_experiment
```

Whether you are a user or a developer, **you are strongly invited to leave untouched the files contained in `code/model/`**. This directory is designed to contain the latest version of the model, which is common to every user at a given time. For every routine that will be specific to `my_experiment`, a new subroutine should be created in `code/my_experiment/src`. This can be achieved using the following command:

```
# Copies the initial conditions subroutine
cp model/src/ini_st.f90 my_experiment/src
```

The compiling step (<**FIX ME: see Section on compiling**>) includes a superseding procedure that will take into account the new version of `ini_st.f90` (in `code/my_experiment/src`) and disregard the standard version found in `code/model/src`. More precisely, it will create the makefile based on this new state of the model (in `./mkfile`), compile and create the executable `code/exe/nh_my_experiment`. More details on `compile.sh` may be found by running:

```
# Provides more information on compile.sh
sh tools/compile.sh -help
```

5.2 Defining your model grid

The grid size is defined in `size.h`. If you wish to modify the grid size, you must first copy `size.h` into your experiment's directory:

```
# Copies the grid file
cp model/inc/size.h my_experiment/inc
```

Grids used in previous experiments are listed in this file and commented out. If your grid appears in a commented line, Comment the uncommented line and uncomment the one you want. Be aware that if your grid set requires more than 2Go, you might experience compilation issues. If so, you may fix the issue by editing `tools/genmakefilel` to replace the default compiling options by:

```
fflags_o="-fpp -real-size 64 -mcmodel medium -shared-intel -stand 03 -u"
fflags_e="-fpp -real-size 64 -mcmodel medium -shared-intel -stand 03 -u"
```

If your grid set does not appear in `my_experiment/inc/size.h`, you can create the required line. Defining the model grid is not straight-forward, because of the multi-grid solver mgrid <**FIX ME: (See Section)**>. The multi-grid solver is used to allow the reuse of array space in `mgrid.f90` <**FIX ME: insert link to function?**>. Although this issue could now be circumvented by making use of f90's dynamic allocation of memory, the code was originally in fortran77, explaining the need for space re-allocation. A step-by-step approach to defining your own grid is provided below:

1. Choose grid dimensions NI, NJ, and NK (i.e., the number of grid cells in x, y, and z directions) such that the grid can be subdivided a maximum

number of times by a factor of 2 to form "$ngrid$" levels of grid. For example, choosing $NI = 48$, $NJ = 24$, and $NK = 32$ constrains the grid levels to 4 (i.e., $ngrid = 4$), because:

$$\begin{aligned} NI &: 48; 24; 12; 6; 3 && \text{(5 grid levels)} \\ NJ &: 24; 12; 6; 3 && \text{(4 grid levels)} \\ NK &: 32; 16; 8; 4; 2 && \text{(5 grid levels)} \end{aligned}$$

The number of grid points possible for a specific $ngrid$ can be computed by multiplying prime numbers (2, 3, 5, 7, etc.) by $2^{ngrid-1}$. Table 1 lists some of the most commonly used number of grid points, depending on the number of grid levels $ngrid$.

2. Compile `tools/preproc.f90`:

```
# Compiles preproc.f90 (e.g., using ifort)
ifort preproc.f90 –o preproc
```

3. Runs `preproc.f90` and fill the values that are asked:

```
# Runs preproc.f90
./preproc.f90
```

4. Copy/Paste the last line the program provides in `my_experiment/inc/size.h`. Below is an example for $NI = 96$, $NJ = 160$, and $NK = 32$ (hence $ngrid = 5$, see Table 1):

```
./preproc
 number of grid levels in mgrid, ngrid =
5
 input the grid info
 NI =
96
 NJ =
160
 NK =
```

Table 1: Number of grid points associated with a specific number of grid levels $ngrid$. These numbers can be computed by multiplying prime numbers (2, 3, 5, 7, etc.) by $2^{ngrid-1}$. Each experiment's number of grid levels is set by the minimum $ngrid$ associated with NI, NJ, and NK.

$ngrid$	Number of grid points (NI, NJ, or NK)							
4	16	24	40	56	88	104	136	152
5	32	48	80	112	176	208	272	304
6	64	96	160	224	352	416	544	608
7	128	192	320	448	704	832	1088	1216

```
32
 Number of grid points on fine grid: nx,ny,nz   96   160   32
m, ntint, ntout, nbc(m)  1      491520      539784      47104
m, ntint, ntout, nbc(m)  2       61440       73800      11776
m, ntint, ntout, nbc(m)  3        7680       10920       2944
m, ntint, ntout, nbc(m)  4         960        1848        736
m, ntint, ntout, nbc(m)  5         120         384        184
Copy the following line to size.h
INTEGER, PARAMETER :: NI=96, NJ=160, NK = 32, ngrid=5, maxout
    =626736, maxint=561720, int1=491520
```

5.3 cppdefs.h

This file defines the different options to be used in the experiment. Again, it is recommended to copy this file into the experiment folder (e.g., `my_experiment/inc/` before making any modifications. To include (exclude) an option, use #define (#undef) $option\_name$. The file includes 13 options:

$runtracmass$: placeholder

$periodic\_ew$: placeholder

$periodic\_ns$: placeholder

$allow\_particle$: If defined, allows the seeding of particles in the experiment. Please refer to section <**FIX ME: ref to particle section**>for a detailed explanation on particle seeding.

$rhoonly$: If defined, only the density field rho is used. The density field is stored in the salinity array (s; see `evalrho_rho.f90`). If not defined, rho is computed from the salinity (s) and temperature (T) fields (see `evalrho_sT.f90`).

$relaxation$: placeholder

$fixed\_bottom\_thickness$: placeholder

$file\_output$: placeholder

$file\_output\_cdf$: placeholder

$file\_output\_bin$: placeholder

$gotm\_call$: placeholder

$implicit$: placeholder

$parallel$: placeholder

5.4 namelist

This file defines key parameters relating to the experiment (e.g., grid resolution, time step, diffusion, output, ...). Again, it is recommended to copy this file into the experiment folder (e.g., `my_experiment/` before making any modifications. Each parameter in the file is either self-explanatory or include a short description as a comment.

5.5 Defining the initial conditions

Initial conditions can be specified either in the corresponding subroutines, or from an input file. The former approach is used in the Shelfbreak test-case (see Section 4.3), where the temperature and salinity distributions are determined from analytical expressions in DO-loops, and only requires a limited knowledge of the model grid. The latter approach can sometimes be more practical, especially when using available data products to initialize the model. However, this approach requires mapping the data used to initialize the experiment to the pre-defined model grid.

The horizontal grid is relatively straightforward to determine, given the grid size specified in `my_experiment/inc/size.h` (i.e., NI and NJ), and the

grid resolution specified in `my_experiment/namelist` (i.e., dx and dy). The location of each grid point can be computed using the following equations:

$$x(i) = -dx/2 + idx; \qquad i = (0, 1, 2 \ldots NI, NI + 1) \tag{1}$$

$$y(j) = -dy/2 + jdy; \qquad j = (0, 1, 2 \ldots NJ, NJ + 1) \tag{2}$$

5.5.1 Temperature, salinity, and density

The initial conditions in the temperature and salinity