Nansen Environmental and Remote Sensing Center

A non-profit environmental research center affiliated with the University of Bergen

Thormøhlensgate 47
N-5006 Bergen, Norway
Tel: +47 55 20 58 00
Fax: +47 55 20 58 01

Technical Report No. XXX

NERSC-HYCOM 2.2

by

Francois Counillon and Knut A. Lisæter

Draft from December 12, 2013

Nansen Environmental and Remote Sensing Center (NERSC)
Thormøhlensgate 47,
N-5006 Bergen
Norway

phone +47 55205800
fax +47 55205801
email admin@nersc.no

TITLE	**REPORT IDENTIFICATION**
NERSC-HYCOM 2.2	NERSC Technical report no.XXX
CLIENT	**CONTRACT NO.**
—	—
CLIENT REFERENCE	**AVAILABILITY**
—	OPEN
INVESTIGATOR	**AUTHORIZATION**
Francois Counillon email: francois.counillon@nersc.no	—

Contents

NERSC-HYCOM 2.2

Francois Counillon and Knut A. Lisæter

Nansen Environmental and Remote Sensing Center

Abstract

This document describes how to use the NERSC-version of HYCOM; e.g. how to set up the : projection, grid, topography, bathymetry, relaxation, nesting etc ...
The main difference of the NERSC-code compare to the standard one, is that it offers more alternatives for ice modelling, possibility to run perturbed ensemble runs, handle atmospheric forcing differently, and allows to run a coupled version with the NORWECOM ecosystem model. The code and scripts are also developed to set-up a model very easily. You just need to follow the step explained in this document. The code is available on subversion so you can easily remain up to date with the latest changes.
The most recent version currently available is version 2.2.37. Please if you observed any bug contact us (francois.counillon@nersc.no). We would be also very pleased to get more contributors to this code-contact us if you are interested.

Chapter 1

Introduction

This manuscript describes the version of HYCOM use at NERSC. HYCOM has been used in NERSC for the past 15 years, starting from MICOM and then moving to HYCOM following the development made within the community. The NERSC version of HYCOM had significantly drifted away from the standard version of HYCOM and used different methods for: nesting, tidal boundary, ice modelling and thermodynamics and forcing field. In 2008, Knut Arild Lisæter initiated a convergence of the NERSC code towards the standard one developed at Stenis. In the present version of NERSC HYCOM the main discrepancies are in: the coupled ice modelling (EVP, MIZ, MIZ+waves, ICESTATE), different way to handle atmospheric forcing fields, allow for running perturbed ensemble run, different river forcing methodology, coupling with NORWECOM ecosystem model ... Last but not least, the code is developed to allow a user friendly usage of the HYCOM code. Numerous scripts, allows you to set up a model grid without to "dig" into the code and set up the code easily. This effort is highly indebted to Knut Arild Lisæter, which developed most of the utility routines and drafted the first version of this document. The effort is currently continued by Francois Counillon, and others - Annette Samuelsen, Vincent Vionnet, Dany Dumont, Intissar Keghouche and Laurent Bertino to name a few. All the code developed at NERSC is separated from the main HYCOM code (when it is not possible CPP FLAGS are used) so that it is easy to maintain our code up-to-date with the most recent version of the HYCOM community. Currently the most recent version of the code is 2.2.37, which is the latest available.

Chapter 2

Subversion: svn

Subversion is a useful tool, to maintain community code. It allows several persons to work on the code at similar time. It is possible to only update a small part of the code but to avoid conflict, you cannot update a file if it is not initially based on the most recent version of the code. You can also check the difference, or the status of each bit of your code. The NERSC Version of HYCOM is put on the nersc server. The svn is accessible to everyone, so if you get trouble to access it, you should contact us. To retrieve the code you need to install the program subversion. On hexagon, the program is already installed and all you have to do is to load the module, type:
module load subversion
To retrieve the whole NERSC-HYCOM code and utilities, type the following command:
svn checkout https://svn.nersc.no/repos/hycom/ .
You will get a hycom folder that contains four subfolders: HYCOM_2.2.12 and HYCOM_2.2.37 and MSCPROGS Doc. The latter corresponds to the latex folder that has been used for creating this pdf. HYCOM_2.2.12 and HYCOM_2.2.37 corresponds to the two latest versions of the NERSC-HYCOM code that have been merged with the standard hycom. MSCPROGS contains utility routines developed at the Mohn Sverdrup Center to plot, compute diagnostic or help setting up a model configuration. Among this two versions of HYCOM available, it is hard to recommended one of them. The most recent often offers newer appealing scheme and proposes correction of older bugs, while the older one is better tested and thus "safer". The writing permission is restricted to few editors, so that you wont be able to commit changes yourself. If you notice a bug that need to be correct you may contact us (francois.counillon@nersc.no) or (Annette.samuelsen@nersc.no) if it is part of the ecosystem model.
We would also be very pleased to have more contributors to this code so if you are interested you may be given an administrator account.
In the following we give you some useful command:

- to update you code, type:
  svn update
  This will update all the part of your code that has not been modified.

- To status of your code, type:
  svn status
  This will return a list of the files that differ from the one saved in the svn. "!" will be reported if the file from the svn is missing, "?" is used if the file is not present in the svn, and "M" is used if the file has been modified.

- To add a file or a folder into the svn, type:
  svn add <filename>

- To remove a file or a folder from the svn, type:
  svn delete <filename>

- To use an older version of the code type:
  svn update -r <version_number>

- To commit a change on the svn, tyoe:
  svn ci -m "Text explaining the change briefly" <files or dir>

Please not that it is not possible to copy a folder from svn and call it differently because svn add a lot of .svn folder into each directory. To remove the annoying .svn file, you can use the following command:
find <initial_folder > | grep -v '/\.' | <destination_folder>
This will copy the intial_folder into destination folder without copying the .svn dir.

The svn is also combined with a program call Trac. This allows you to browse the code, check the wiki of the code, the Timeline of the change (you can visualise the change, search for a key word, ...) on the following website:
https://svn.nersc.no/hycom/timeline
There is also a section called ticket, this corresponds to known bug in the code, that are not yet solved, but we have not got the time to solve it.

Chapter 3

Model Setup

This chapter deals with the setup of the HYCOM model, using shell scripts and FORTRAN programs developed at NERSC. This chapter only deals with setup of data/forcing files, for compilation of the model code see Chapter 4.2.

In this new setup, the NERSC version of HYCOM use a closer framework and code to the standard setup of HYCOM(**?**), than in previous version. The benefit of this is that it should be easier to use the tools present in the standard HYCOM distribution, and to exchange our tools with other groups. The new framework is also safer, as it avoid mixing up topography, parameter etc ... The new setup collects all the different routines and data used to drive the model in one central directory for each model region. Within this directory there are separate subdirectories for the model topography, atmospheric forcing data, relaxation data as commonly used in standard HYCOM. Within these subdirectories, shell scripts are used to set up the data calling programs from the standard and NERSC HYCOM distribution. Finally, there are also subdirectories for handling the different experiments.

The general guidelines of setting up a model and a region are given in this document. NB: experienced user can directly jump on the checklist in Section A. It gives step-by-step instructions on how to set up the model.

3.1 The top-level region directory

The top-level region directory is the directory where all the input data needed to run HYCOM is kept and created. A top-level region directory is tied to the geographical location of the model grid and its horizontal resolution. When the placement of a model grid is determined (typically using the conformal mapping routines), a region directory for the grid is created at a given horizontal resolution. The contents of the top-level directory will typically look something like this:

```
drwxr-s---   4 fanf mohnsv     4096 2011-07-12 15:34 bin
drwxr-s---   4 fanf mohnsv     4096 2011-07-12 15:34 force
drwxr-s---   4 fanf mohnsv     4096 2011-07-12 15:34 relax
drwxr-s---   4 fanf mohnsv     4096 2011-07-12 15:34 topo
drwxr-x---   3 fanf mohnsv     4096 2011-07-12 15:34 nest_nersc
drwxr-x---   3 fanf mohnsv     4096 2011-07-12 15:34 other_nersc
drwxr-x---   3 fanf mohnsv     4096 2011-07-12 15:34 tides_nersc
drwxr-s---   4 fanf mohnsv     4096 2011-07-12 15:34 expt_01.0
drwxr-s---   4 fanf mohnsv     4096 2011-07-12 15:34 expt_01.1
-rw-r-----   1 fanf mohnsv     5632008 2008-09-11 19:17 REGION.src
lrwxrwxrwx   1 fanf mohnsv     52 2011-07-12 15:34 config
lrwxrwxrwx   1 fanf mohnsv     56 2011-07-12 15:34 src_2.2.12
```

In this listing, there are subdirectories for the data needed to run the model as in the standard HYCOM (here **topo/**, **force/**, and **relax/**), specific for the NERSC implementation (here **nest_nersc/**, **tides_nersc/**, and **other_nersc/**) as well as experiment directories (here the directories **exp_01.0** and **exp_01.1**). The file **REGION.src** is a text file containing some definitions for the setup of the hycom model, and it will be used by many of the scripts for setting up the forcing.

3.1.1 Setting up a new region

When a new region needs to be set up, the first thing to do is to decide on a suitable name. This name will be used both to name some of the data files used by HYCOM, as well as naming the region directory itself. The name of the region directory should be something descriptive of the region - as an example, take the TOPAZ3 grid, a suggested name of the top-level directory in this case is "TP3a0.12" - where TP3 is a three-letter ID of the grid (here based on the name "TOPAZ3"). The letter a is a version number reflecting *minor* changes to the grid (a large change may warrant a new ID), and 0.12 is descriptive of the resolution, here roughly 0.12 degrees. The name will be referred to as the region name in the following.

When a suitable name is decided upon, a new region directory should be created. The easiest way of doing this is to start with an already existing region directory. It is recommended to keep one "template" region on which to base other regions on. This way, updates/modifications to existing setup scripts can be done by just modifying the main "template" directory. On the svn, a "template" of a North Atlantic configuration with 1° resolution is given:

./hycom/HYCOM_2.2.12/NATa1.00.

A shell script exists for setting up the architecture of your new region:

./hycom/HYCOM_2.2.12/NATa1.00/bin/newRegion.sh

Two arguments are required. The first is the region name, the second is the path where you want to place the region on the file system. As for example, the following command

./hycom/HYCOM_2.2.12/NATa1.00/bin/newregion.sh TP3a0.12 /work/fanf/

will place the new region directory **TP3a0.12/** in the work directory **/work/fanf/**. The script will copy all files to the new top-level region directory, except the data files that are not valid for the new region - these must be set up by you and this is described later in this chapter. In addition this script will attempt to modify the setup file **REGION.src** for the new region. At first you must replace the path of MSCPROGS with your own compiled as explained in Chapter 6. (by default it assumes that the path is ∼/hycom/MSCPROGS) You must also verify that:

- The path given in **REGION.src** exists. Whenever the model is moved to another machine, these paths must be replaced to location of the corresponding datasets.
- The variable "R" should be set to the region name described earlier. (This should be done automatically by the script **newRegion.sh**)
- The environment variables HYCOM_ALL exists and is properly set. It corresponds to configuration script from the standard HYCOM. This need to be compiled only once by the first user on the machine, so the path may not be on your account. In hexagon, this code can be found in **/home/nersc/fanf/ALL**

The following step for setting up a new region is to copy the various grid-related files and the topography files into the topo directory. This is explained in the next Section.

NB: Now that the region directory has been set up, it will also contain all the routines which are used in the rest of this chapter. When a script is mentioned in the setup procedures later on, you should be able to find it somewhere inside the region directory.

3.2 Topography

All file related to the topography will be placed/produced in the directory **topo/** under the region directory. Unlike the model grid location (i.e. the regional.grid.[a,b]), there can be several versions of the model topography for a region. For instance, there can be a topography version for an un-nested model, a topography for nesting the model from an outer model "A", and a topography for nesting the model from an outer model "B". Another example is if you want to modify an existing topography to improve model performance, such as making sure the bathymetry of the Denmark Strait is sufficiently deep.

Standard HYCOM has something called a "topography version", which is used to keep track of the different topographies you create for a model grid. Whenever you modify a topography it should be saved as a different topography version. You should make sure you don't overwrite an existing topography. Here is an example of a "versioned" topography folder of *topo/*

```
drwxr-s---  4 knutali mohnsv     4096 2011-07-12 15:34 bin
-rw-r-----  1 knutali mohnsv  5632008 2008-09-11 19:17 depths800x880.uf
-rw-r-----  1 knutali mohnsv  2818048 2008-09-11 20:22 depth_TP3a0.12_00.a
-rw-r-----  1 knutali mohnsv       85 2008-09-11 20:22 depth_TP3a0.12_00.b
-rw-r-----  1 knutali mohnsv  2818048 2008-09-11 20:22 depth_TP3a0.12_01.a
-rw-r-----  1 knutali mohnsv       85 2008-09-11 20:22 depth_TP3a0.12_01.b
-rw-r-----  1 knutali mohnsv      569 2008-09-11 12:57 grid.info
drwxr-s---  2 knutali mohnsv     4096 2008-09-11 20:24 partit
-rw-r-----  1 knutali mohnsv 53542912 2008-09-11 12:57 regional.grid.a
-rw-r-----  1 knutali mohnsv     1174 2008-09-11 12:57 regional.grid.b
```

This directory shows two topography "ab" file pairs:
depth_TP3a0.12_00.[ab] and **depth_TP3a0.12_01.[ab]**.
The naming convention of topography files is **depth_$R_$T.[ab]**, where $R is the name of the region (example TP3a0.12) and $T is the topography version ($T). When HYCOM will be run with a given topography version, the corresponding depth file will be copied to regional.depth.[a,b] in the running directory.

3.2.1 Importing a model topography

A script located in the topo/bin directory allows to import topographies created by tools described in Section 3.2 into the region structure.

- Decide on a topography version number T. The following convention is recommended for $T:
  - The topo number for files directly output from ETOPO, GEBCO is 00
  - The topo number for files directly output from the grid program after topofix is 01
  - The topo number for topography files modified for nesting purpose is >= 2. The motivation is that if you want to nest your model into another outer model you want to start from topography 01 to avoid multiple interpolation.
- Locate the directory containing the topography files that you want to import in the region/topo directory.
- Run the **regioncopy.sh** script, as follows: **region/topo/bin/regioncopy.sh 01 /work/knutali/topofiles** where 01 is the chosen topography number, and **/work/knutali/topofiles** is the place where the new topography files that you want to import are located. Beware of the possible pitfalls of this routine, see Section 3.2.3 in case of trouble or disturbing warning messages. This script copy the grid files (if they are not present):
  1. regional.grid.[ab]
  2. latlon.dat
  3. newpos.uf

The files **newpos.uf** and **latlon.dat** are deprecated in the newer version as they contain recurrent info than the regional.grid.[a,b]. However they are still copied as some "old-routine" may still need them.

Depth files are also copied and it is ensured that they are compatible with the format of HYCOM 2.2 (see addhuge.py bellow). The script choose one depth file and use this priority order when converting from the old topography files:

1. regional.depth.[ab]
2. ndepths*.uf
3. depths*.uf

Similarly **depths???x???.uf** and **ndepths???x???.uf** correponds to the previous version of HYCOM (version 2.1). This script can support them but regional.depth.[ab]-file is safest and prioritised. You should be careful if you consider using them. Usually, the ndepths corresponds to the depths that have been changed by nestbat, meaning that the topography has been interpolated to match a coarser model. It is not recommended to use bathymetry that have been interpolated several times. Ideally, We recommend to restart the grid creation is you have any doubt.

3.2.2 Create a nested model topography

To set up a nested model topography, follow this recipe:

- Decide on a new topography version number T, and the topography version to use as basis for the new topography.
- Locate the directory containing the outer model region.
- Run the **nestbat.sh** script to run the routines which smooth the inner model topography towards that of the outer model.

When nesting a model into coarser one, try to follow this basic recommendations:

- Put the boundary far enough of the circulation feature you are interested in (approximately 30 grid cells).
- Nesting need to applied on a buffer zone to handle topography difference, viscosity difference etc . (approximately grid cell). Tools used in nestbat will interpolate the inner model topography in a buffer zone. The first point the buffer zone is equal to the outer model and the last one is equal to the bathymetry of the inner.
- Ratio between the outer and the inner should be around 1:3 - 1:5. Larger is not recommended and may lead to model instabilities. However, stability can be reached by extending the relaxation buffer zone, and forcing artificially the relaxation strength.
- In order to solve the hyperbolic equation, the 3 first grid cells are used. If the coastline is not uniform (e.g. an island in the second grid cell) within them, this may caused some instabilities and slow down considerably the model running speed.

3.2.3 Help routines for setting up topo-files

Following is a list of script located in topo/bin that are not necessary for setting up a model, but that may be useful for advanced users. Most of them are already called in regioncopy.sh and nestbat.sh

toponame.sh Short script for getting the topography name for a given topography version.

addhuge.py The older versions of NERSC HYCOM used a zero value to denote land. Newer versions will use the value 2.0^{99} to denote land (following standard HYCOM). This script replaces zero values with 2.0^{99} in the standard HYCOM topography files. The script is run by the regioncopy.sh script and is rarely needed to be run by hand. The python path may have to be changed as the file is migrated between different machines.

depthtoregional.py This routine will convert a old depths*.uf (or ndepths*.uf) file to a topography file. The script is run by the regioncopy.sh script and is rarely needed to be run by hand. The python path may have to be changed as the file is migrated between different machines.

The python scripts (.py suffix) depend on the location of the locally installed python. If the script stops with strange error messages, you should check that the first line in the python scripts point to an actually existing python installation.

Variable	Description
X	Experiment number - a number of the type AA.B, an example is 01.0
E	Based on $X - a number of the type AAB, an example is 010
T	Topography version - can be tied to an experiment.

Table 3.1: Environment variables set by the file **EXPT.src**. These are used by most scripts which set up HYCOM.

3.3 The experiment directories

Each experiment directory contains a configuration files related to the experiment **EXPT.src**, model parametrisation setting files **blkdat.input**, **infile.in**, **infile.evp**, **infile.icestate** and, **ports.input**, and job script for launching job on supercomputers **preprocess.sh**, **postprocess.sh**, **pbsjob.sh**. The forcing, the topography and the relaxation files are located at the root of the region directory the **force/**, **topo/**, **relax/**. The useful files are copied before a job is submitted depending on the setting of **EXPT.src**.

All aspects of the model can be changed from experiment to experiment, except the placement of the model grid and the horizontal resolution (given through files **topo/regional.grid.**[**ab**]). This means for instance that experiments can have a different number of vertical layers, different parametrizations, different code versions, and even different topographies.

The naming of the experiment subdirectories is based on experiment number X which is a floating-point number of the form AA.B, for instance 01.0, 01.1, 01.2 etc. The full name of the experiment directories is **expt_X/** where X would be the experiment number. Again, examples of experiment directories are **expt_01.0/** corresponding to experiment 01.0, **expt_01.1/** corresponding to experiment 01.1 and so on.

3.3.1 Setting up experiments

When you have run the script regioncopy.sh in previous Section, a directory **expt_01.0/** has been created. It contains all necessary files, but some files in the experiment directory need to be properly set up before creating the relaxation, the climatology and the forcing fields some . It consists of the model bathymetry and the vertical density structure (number of layer and target densities). Such info is defined in **EXPT.src** and **blkdat.input**. Following is a brief description of this two files and advices on how to configure them depending of your need.

EXPT.src is the main experiment setting file. It set the model topography version used together with some other environment variables (see Table 3.1). The first variable represents the experiment number that matches the directory name. If the experiment directory is named **expt_01.0/**, the variable X should be set to "01.0". The variable E is derived from X by removing the period. Roughly speaking, $E = X \times 10$. The variable T corresponds to the topography number as explained in Section 3.2.2.

Following is an example of these variables in EXPT.src file:

```
X="01.0"                                # X based on dir name (expt_$X)
E="010"                                 # E is X without "."
T="01"                                  # Topography version
```

this basically states that we are in expt_01.0 and that the topography version is number "01". Note that by convention it is recommended that the two first digits of X is matching the topography number T.

Finally, there is a variable S which corresponds to the location of the scratch directory where files will be moved to before running HYCOM. This place is just temporary and can be deleted if wanted once the hycom run is finished. In the supercomputer hexagon, it should look like that:

export S=/work/$USER/hycom/$R/expt_$X/SCRATCH

The other file that need to be edited is **blkdat.input**. It contains the parametrization of the model such as number of layer, target densities, viscosity, diffusion ... Therefore, this file is quite complex and contains many variable. A description of **blkdat.input** is provided in the HYCOM Users Manual (**?**) and some additional advice are provided in section 5.1.1. At this stage, only variables in Table 3.2 must be set properly before running any of the setup scripts later described in this sections. Once you have specify the idm, jdm, iexpt and iversn, you need to specify the vertical structure of your HYCOM run.

First, you need to decide on a number of vertical layer, *kdm*. Due to its hybrid coordinate properties, HYCOM allows to have less vertical layer than common models (in z-layer or sigma). HYCOM will have high vertical resolution in the mixed layer as the layer will be consider as z-coordinate; and in the interior, the model will have a higher discretisation in place of large density gradient thanks to the isopycnal properties. Thus, it was long thought that just 22 layers would be sufficient, but recent tests show that large benefit are obtained by increasing this number. By default, we recommend 28 layers.

Second, you need chose how many layers will be hybrid (can transit from isopycnal to z or σ-layers). If the number of hybrid is 0 then the model will be fully isopycnal (as in MICOM). Using hybrid coordinate is very beneficial for realistic simulation as you will get a better representation of the mix layer depths. Using only isopycnal layers can be useful in idealised framework, as they allow conservation of T,S and potential vorticity anomaly and allows to study area of development/intensification of instabilities. This option is also chosen in climate simulation where conservation of T and S is mandatory. You can also chose how many layer you allow to transit to σ-layers in shallow water. This is beneficial if you are interested in coastal area. Remember that σ-layers are considered as hybrid so the number of *nsigma* <=*nhybrd*+ .Finally there is an option to have all layer as z. Set nsigma=0.

Third, you need to chose to define of a reference pressure flag. Sigma-0 means that you put the reference at the surface, and Sigma-2 at 2000 m. Problem is that these two set up will become inaccurate the further away they are from reference pressure. Sigma-0 will provide better result near the surface and Sigma-2 in the deeper part of the model. Beware that none of these configurations can handle thermobaricity, which is useful to resolve some processes such as the intrusion of Antarctic Bottom Water in the North Atlantic, or the bottom water flowing out of the Mediterranean Sea. A solution is proposed in the standard HYCOM referred as Sigma-2* (set thflag=2 and kapref accordingly). This implementation is not yet tested in the NERSC-framework, as you need reference state that vary locally depending on the area you are in (Mediterranean, NA, ...).

Fourth, you need to set the reference target densities (from 1–kdm) and thbase. One needs to add 1000 to obtain the volumic mass in kg.m^{-3}. These values should be set depending on the water masses present in the area. You must cover the full range, and if possible target known water masses density. Depending on how far away you are from the reference density, the model will decide in which vertical coordinate you are (isopycnal or z-layer). Note that in TOPAZ we use unrealistically low reference density near the surface, to ensure that the top layers remain in z–coordinate. This was motivated to make sure that some z-layer are present for ecosystem model.

Finally, you need to set the stratification within the mixed layer depth and in the bottom layer (if sigma layers are used). The depth of the mix layer is computed from mixing scheme, e.g. GISS, MY, KPP . Given a certain number of HYCOM z-layer available, HYCOM will organise them depending on dp00, dp00x, dp00f (resp. ds00, ds00x, ds00f for sigma layer) . The first layer will have the minimum thickness (dp00), and their size will increase (by dp00x) until it reaches the maximum thickness allowed (dp00f). If these parameters are badly set (for example dp00f too small), the last available layer may be artificially very large causing artificial spurious diapycnal mixing. This was observed in the Supbolar Gyre, where the mix layer depth reaches 3000 m deep, and dp00f was set too small. the last layer of the mix layer depth was > 1500 m.

NB: The shell script **bin/newExpt.sh** under the region directory will assist in setting up a *new* experiment. It should set the bare minimum of variables in **blkdat.input** and **EXPT.src**.

Note that if you decide to change the topography or the vertical stratification you will need to reprocess all the forcings, clim and relaxation files. Therefore, it is safer to just create a new experiment as explained in the next section.

3.3.2 When to set up new experiments

This is entirely up to you when to set up a new experiment. Some may consider that a changing viscosity is sufficient although it is not strictly necessary. There are a few situations when you MUST set-up a new experiment:

- Changing topography version. If you don't change experiment for this there will be a inconsistency between model depth in restart files (pbot) and topography files. There will also be inconsistencies between relaxation fields and topography files.

Variable	Description
idm	1st grid dimension, must match idm in **regional.grid.b**
jdm	2nd grid dimension, must match jdm in **regional.grid.b**
iexpt	Must match experiment variable "E" in **EXPT.src**
iversn	Must match hycom version (e.g 2.2.12 means iversn = 22)
kdm	Number of layers in the vertical
nhybrd	Number of hybrid levels (0=all isopycnal)
nsigma	number of sigma levels (nhybrd-nsigma z-levels)
dp00	z-level spacing minimum thickness (m)
dp00x	z-level spacing maximum thickness (m)
dp00f	z-level spacing stretching factor (1.0=const.space)
ds00	shallow z-level spacing minimum thickness (m)
ds00x	shallow z-level spacing maximum thickness (m)
ds00f	shallow z-level spacing stretching factor (1.0=const.space)
locsig	locally-referenced pot. density for stability (0=F,1=T)
kapref	thermobaric ref. state (-1=input,0=none,1,2,3=constant)
thflag	reference pressure flag (0=Sigma-0, 2=Sigma-2)
thbase	reference density (sigma units)
vsigma	spacially varying isopycnal target densities (0)
sigma layer 1	target reference density of the first layer (sigma units)
sigma layer 2	target reference density of the first layer (sigma units)

Table 3.2: The bare minimum of variables to set in the file at this stage **blkdat.input**

- Changing the vertical grid , i.e. variable in Table 3.2. If you don't set up new experiment for this, there will be a inconsistency between the relaxation fields or the climatology with model fields.
- Changing hycom versions. If the HYCOM version is updated, there are usually changes to parameters in **blkdat.input**. Creating a new experiment directory you can make the necessary changes to **blkdat.input**, without rendering the old **blkdat.input** useless for running the old model...

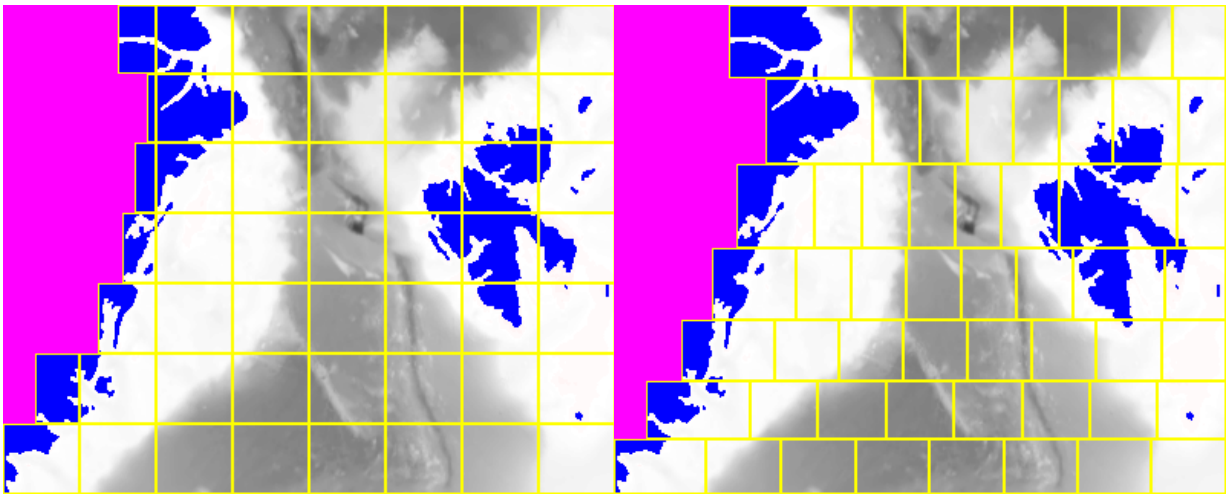
3.3.3 How to create/copy and backup experiment

After creating a new region, a first template of experiment (named expt_01.0) is placed the region directory. A whole bunch of scripts are available to handle experiment easily in region/bin/.

newExp.sh copy the basic structure and necessary files for the experiment from an existing one and changes the EXPT.src. You will still need to re-create the relaxation, climatology and forcing files. this script is used as follow:
newExp.sh old_experiment_number new_experiment_number

cleanEXP.sh allows you to delete all the file related to an experiment: model output, relaxation, climatology and forcing files. this script is used as follow:
bin/cleanEXP.sh experiment_number

backup_exp.sh allows you to backup all useful files related to an experiment: model output, relaxation, climatology and forcing files, nesting files and the topo files. Temporary files are not saved (i.e. the SCRATCH directory). The script create a file name called backup_expt*X*.tar.gz where X is the experiment number and save it in the location *BACKUP_PATH/REGION_NAME* where BACKUP_PATH and REGION_NAME ($R) are set in the REGION.src. The script uses as follows:
backup_exp.sh *X*
To recover an experiment you then just need to copy the backup file to the root of your region and then gunzip and detar it.



(a) Uniform partitioning of domain (b) Non-uniform partitioning of domain

Figure 3.1: Examples of two different MPI partitions of a model domain

3.3.4 Generating MPI partitions

In order to run a model using the Message Passing Interface (MPI), it is necessary to specify how the model domain will be split into subdomains. A separate routine from the standard HYCOM distribution is designed to do this, and it is possible to call it through the shell script **topo/bin/tile_grid.sh**. This routine will create a text file describing the partition (example: depth_TP31.2_01.0016 for 16 tiles) and a picture of the partitioning (example: depth_GH01.2_01.0016.ppm) that are placed inside the **topo/partit/** subdirectory. If the display program is available, the script will open the picture (on hexagon, this can be done by the command 'module load imagemagick-login').

The **tile_grid.sh** routine takes as input the number of subdomains along the two dimensions of the model grid, along with the topography version it will do the partitioning on. This means that you need to create MPI tiles for all the topography versions you use.

The routine **tile_grid.sh** can be called in two modes, with positive or negative arguments. For example:

- bin/tile_grid.sh 8 7 01
- bin/tile_grid.sh -8 -7 01

The result of the two different partitioning methods is illustrated in Figure 3.1.

With negative arguments, figure 3.1a, the partitioning will be uniform - the size of each tile will be (approximately) the same. This setup gives the lowest memory footprint when running the model. On the other hand the workload between the MPI tiles is not evenly balanced, since some tiles will have more sea points than others. With positive arguments, figure 3.1b, the partitioning is non-uniform, but the number of sea points on each tile is approximately the same. The tiling balances well the work between MPI tiles meaning that the MPI waiting time will be reduced, but the memory footprint of the model will be higher since the total memory usage depends on the *maximum* size of all the tiles.

The optimal partitioning depends on the model and the machine, but here are some general guidelines

1. choose a uniform partitioning (negative values) when memory is an issue
2. choose a non-uniform partitioning (positive values) when the total number of MPI tasks is low
3. choose a uniform partitioning when the total number of MPI tasks is high, as the communication seems to be more efficient in this case

Remember that these are guidelines, and will probably depend on the model grid. Some degree of trial and error is always needed to find the optimal setup. Also note that memory per MPI tile often becomes an issue when the number of MPI tasks is low, giving some inconsistency between point 1 and 2 above.

NB: Note that the choice of domain partitioning is not strictly necessary before you start compiling the model, see Section 4.2.

3.3.5 Precribed barotropic inflow: ports.input

In the standard HYCOM, the barotropic boundary condition are handle using a file called ports.input. There are to ways to force an inflow at a boundary: using input from an outer model (: i.e. filelbflag is equal to 2 in blkdat.input) or using a prescribed inflow based on theory (i.e. filelbflag is equal to 1 in blkdat.input). In this section we will focus on the later. For example in TOPAZ, we would like to consider the inflow from the Pacific through the Bering Straits, but no outer model is available. Note that here we only consider the barotropic boundary condition (Barotropic velocities and sea level), the baroclinic is handle using relaxation. This is addressed either in the next section (if you want to relax the water to climatology) or section 3.6 if you want to relax the condition to an outer model.

Barotropic condition in HYCOM are treated with a flather technique. The 3 first points are used to solve the hyperbolic equation.The way this is handle in HYCOM is crude and the coastline must be uniform in the 3 first grid cells otherwise large instabilities will appear. You should make sure that there is no island, or change in the coast line within these grid cell (try even to do that for a couple more: e.g. 5).

To activate a prescribed barotropic inflow in HYCOM, you need to first set **lbflag** is equal to 1 in blkdat.input. This mean that HYCOM will look for a file call ports.input where the specificities of the flow are detailed. This file should be place in the rots of the experiment directory. Second you need to edit this file: A typical ports.input with two ports will look like that:

```
  2                                                 'nports   ' = number of ports
  1                                                 'pefold   ' = port transport e-folding time i
  4                                                 'kdport   ' = port orientation (1=N, 2=S, 3=E
222                                                 'ifport   ' = first i-index
222                                                 'ilport   ' = last  i-index (=ifport for east
813                                                 'jfport   ' = first j-index
836                                                 'jlport   ' = last  j-index (=jfport for nort
  0.0                                               'svpnow   ' = existing port transport in Sv (
  0.7                                               'svport   ' = target   port transport in Sv (
  3                                                 'kdport   ' = port orientation (1=N, 2=S, 3=E
800                                                 'ifport   ' = first i-index
800                                                 'ilport   ' = last  i-index (=ifport for east
  2                                                 'jfport   ' = first j-index
 30                                                 'jlport   ' = last  j-index (=jfport for nort
  0.0                                               'svpnow   ' = existing port transport in Sv (
  0.7                                               'svport   ' = target   port transport in Sv (
```

The first line will tell you how many ports (Barotropic inflow) you are going to impose. The second line is the e-folding time, and should not be changed. Then begin the description of each port:

- Need to specify a box area where the flow will be imposed (ifport, ilport, jfport,jlport)
- Need to specify on which side of the box, the flow will be imposed (kdport)
- Specify how much transport you want to have (svport)

The variable svpnow is not used. There are some option to provide transport that varies seasonally or inter-annually, see Section . If the code is compiles with a flag SEASONAL_PORT or INTERANNUAL_PORT. In the former case, the model will look for a ASCII file called clim_tran.txt located in REGIONDIR/relax/exptnumber/, which contains daily estimate of transport. In the later case, the model will look for in the same directory a file corresponding to the running year (example: cm_tran_1980.txt if the model is run in 1980).

3.4 Relaxation

Relaxation fields can be used for sidewall/surface nudging and for initializing the model. Currently the climatologies is based either on Levitus or on PHC climatologies. Levitus corresponds to the World of Atlas 2005 climatology while PHC is the version 3.0 of the Polar Science Center Hydrographic Climatology. The latter is recommended if the Arctic is within the model domain. Again the topography and the vertical

structure of the model edited in Section 3.3.1 must now on never be changed. A two-step procedure is needed to set up the relaxation fields. First, a horizontal interpolation on the z-levels of the climatology, (a step which depends on the model grid location). The final step is a vertical interpolation from z-levels to the isopycnal levels, which depends on the setup of hybrid and isopycnal layers of the model. The routines which are used to do these interpolation steps are from the **HYCOM_ALL** directory which is supplied with the standard HYCOM distribution.

The relaxation fields are set up in the subdirectory **relax/**. A typical listing of the contents of that directory may look like this

```
drwxr-s---  3 knutali mohnsv 4096 2008-11-24 20:15 010
drwxr-s---  3 knutali mohnsv 4096 2008-11-24 20:19 bin
drwx--S---  4 knutali mohnsv 4096 2008-11-24 19:48 levitus
drwx--S---  3 knutali mohnsv 4096 2008-11-24 19:36 phc
-rw-r-----  1 knutali mohnsv 1808 2008-11-24 20:09 README.KAL
-rw-------  1 knutali mohnsv 3580 2008-09-11 10:27 README.relax
```

levitus holds Levitus climatologies at predefined isopotential(z) surfaces interpolated horizontally on the model grid.

phc holds phc climatologies at predefined isopotential(z) surfaces interpolated horizontally on the model grid.

bin This directory contains routines for creating the climatologies.

010 This contains the final relaxation files for experiment 01.0. Each final files contains monthly climatology interpolated into the model grid. It consists of the temperature (relax_tem.[a,b]), salinity (relax_sal.[a,b]), layer interface (relax_int.[a,b]), and relaxation coefficients (relax_rmu.[a,b]).

3.4.1 Routines for generating relaxation fields

The routines in the *relax/bin/* directory can be used to set up the relaxation fields. This routines are called with the experiment number and retrieve the information about vertical structure and topography automatically.

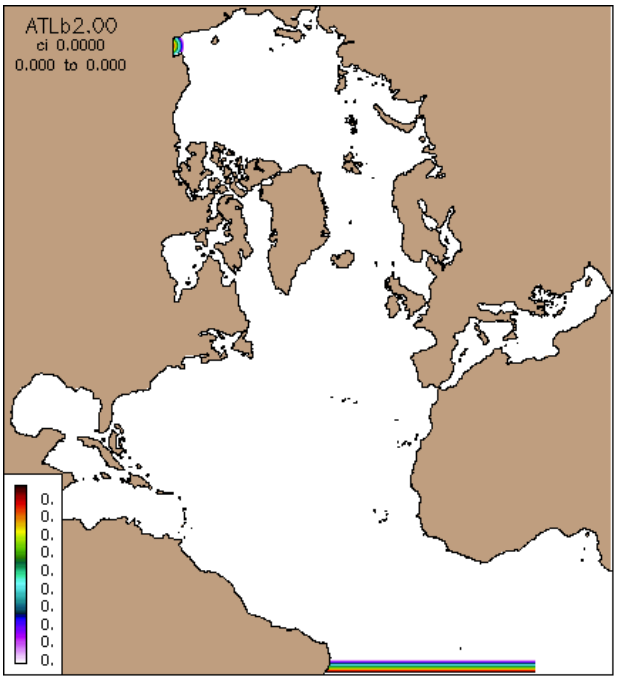
z_levitus.sh creates horizontally interpolated fields on the original Levitus z-levels. Input is "ksigma" which is the same as the "thflag" flag in the **blkdat.input** (i.e. sigma 0 or 2) used by the experiment you want to create climatologies for. The fields created by this routine are dumped in the *levitus/* subdirectory. The Levitus fields used in this procedure can be retrieved from the HYCOM ftp site.

z_phc.sh does the same as for z_levitus.sh, but uses the PHC climatology. The fields created by this routine are dumped in the *phc/* subdirectory. The PHC fields used in this procedure can be retrieved from the HYCOM ftp site.

relax.sh creates a climatology interpolated to the density layers of a given experiment. The input for this routine is the experiment number and the climatology to use(phc or levitus). The actual density surfaces for this experiment will be retrieved automatically from the **blkdat.input** file. This routine requires **z_levitus.sh** or **z_phc.sh** to be run first. The climatology is dumped to subdirectory corresponding to the variable "E" in the experiment file **EXPT.src** (example - 010).

relax_rmu.sh computes the relaxation coefficient for the lateral boundaries and/or for the ports (barotropic inflow, see subsection 3.6). For the latter, you may want that the water flowing into the domain is equal to the climatology. The relaxation coefficient are stored in relax_rmu.[a,b]. The script is user interactive in order to create in a first part an input file called rmu.in. You will be asked if you want to include a port in you model that is not on a model relaxed boundary. If you don't have any port or if it is on the model grid boundary, just answer F, and keep answering the script (Note that a relaxation width of 20 grid cell and a relaxation time scale of 20 days is recommended), That its.

If the answer was true- i.e. that you have a port not at the model lateral boundary (as the Bering Strait in the Figure), then you will need to edit the file rmu.in by yourself. The script will stop and copy of a template at you REGION root directory. Once the file is edited and placed in the experiment



(a) Example of relaxation mask with the given example of rmu.in. Red areas are equal to 1/20.

subdirectory. You can then relaunch the script answer .true. once more and the corresponding binary file will be created. The rmu.in file will typically look like that:

```
F                    # Flag for relaxation on Eastern  boundary
T                    # Flag for relaxation on Western  boundary
T                    # Flag for relaxation on Southern boundary
T                    # Flag for relaxation on Northern boundary
20                   # Boundary relaxation length (grid cells)
20                   # Point relaxation length (grid cells)
20                   # Time relax
BER T -167.0 66.5    # Bering Sea point
END T    0.0  0.0    # !!!!! Ends point specification
700 800   1 1   F    # Specifies rectangle boundary flags (Switches off (F) relaxation in t
```

The first four lines indicates if you want to have relaxation along each model boundary (T/F). Then the fifth line indicates on how many grid cell the relaxation should occurred for lateral boundary relaxation (strongest on the first grid cell and decreasing). The sixth line is the same but for given point relaxation (i.e. for ports defined). If your point is on a relaxed boundary, it would not change anything. For point relaxation, the mask is circular. The next line corresponds to the relaxation time scale, which means the number of days it takes for having the grid cell matching the relaxation value. Then follow a list of point relaxation that consists of a tag name, a boolean (T/F), followed by geographical coordinate and # sign to indicate the description of the file. Is it quite unpractical to have to provide coordinate and not pivot point (unlike in ports.input). There is a tool in the MSCPROGS/GridToLL/gridtoll see Section **??** to extract the lon,lat from model grid point. It is usually common to place the relaxation centred in your port and make sure that the mask cover well the ports. Such list must be finished by the 10th lines. Finally, you can also remove the relaxation in a given area as in the figure with the last line. This example will provide the solution shown in Figure **??**. This is just and example, as usually you want the relaxation to cover the whole boundary.

old_lev_nersc.sh Creates some data files needed when running hycom with the NERSC "old" climatology option - which is generally deprecated..

3.4.2 Prerequisites for generating relaxation fields

To generate the climatologies data files for PHC or Levitus are needed, which can be acquired from the HYCOM ftp site:

ftp://hycom.rsmas.miami.edu/awall/hycom/

The topography files and the file **blkdat.input** must have been set up for the experiment which relaxation fields will be created for.

In addition the environment variables HYCOM_ALL, LEVITUS_PATH and PHC_PATH needs to be set up, this is done in the file **REGION.src**. The variable HYCOM_ALL points to the path of the hycom *ALL* direcory, while LEVITUS_PATH and PHC_PATH point to the location of the Levitus and PHC climatologies, respectively. These variables are also set in **REGION.src**.

In addition, to use the "old" climatology option, the path to the Mohn-Sverdrup programs should be set in the variable MSCPROGS. This variable should also be set up in the file **REGION.src**.

3.4.3 Procedure for generating relaxation fields

To create climatologies this is the basic procedure to follow

1. Create the horizontally interpolated climatologies using the routines **bin/z_levitus.sh** or **bin/z_phc.sh**. This step is experiment-independent just needs to be run once. the PHC climatology is generally recommended, especially if the Arctic is in the model domain.

2. Create the relaxation fields with the script **bin/relax.sh**

3. Create the relaxation masks with the script **bin/relax_rmu.sh**

3.5 Forcing

The forcing directory contains various input fields which are used to run the model. In this directory it is possible to create atmospheric forcing fields (winds, air temperature, radiative fluxes etc), river forcing fields, and so on. The contents of this directory will typically look something like this

```
drwxr-s---  4 knutali mohnsv 4096 Jan  3 19:45 bin
drwxr-s---  4 knutali mohnsv 4096 Jan  3 20:12 nersc_era40
drwxr-s---  2 knutali mohnsv 4096 Jan  3 20:15 other
drwxr-s---  4 knutali mohnsv 4096 Jan  3 20:14 rivers
drwxr-s---  3 knutali mohnsv 4096 Jan  3 20:18 seawifs
```

The **bin** directory contain script for generating the forcing, **nersc_era40** contain climatological atmospherical forcing, **rivers** contains the rivers output forcing, and seawifs represent the climatology of water type used in HYCOM for computing the light penetration coefficient.

The routines in the **force/bin** directory are:

nersc_rivers.sh set up river forcing using point data supplied in a **rivers.dat** file. This file must be present in the force subdirectory (REGION/force). The main routine called is **rivers**, briefly described in Section 6.4.33. If the rivers are few and their flux well monitored, this is probably the best solution, otherwise, use the nersc_trip.sh to generate the river forcing.

nersc_trip.sh set up river forcing using runoff data from ERA-interim (or ER40 with a fix if wanted) + the TRIP05 database to direct the runoff data towards rivers. Section 6.4.39 has a brief description of the routines used.

nersc_clim.sh This routine can be used to set up atmospheric climatology forcing fields. Available climatologies are "era40", "ncepr". Climatology is sometimes used to complete synoptic forcing.

nersc_synoptic.sh This routine can be used to set up synoptic atmospheric forcing. It uses input from the file **infile.in** in the experiment subdirectory to determine the synoptic and climatology data sets to use, as well as the time period to create the forcing for. The main routine used is **forfun_nersc-2.2**, described in Section 6.4.36.

Variable	Description	Available period
SEAWIFS	SEAWIFS data set, supplied by standard HYCOM	monthly climatology
NCEP_CLIM_PATH	NCEP clim produced at NERSC (1.915°,NetCDF)	monthly climatology
ERA40_CLIM_PATH	ERA40 clim produced at NERSC (1.118°,NetCDF)	monthly climatology
ERA40_PATH	ERA40 data set (1.118°,NetCDF)	1957-2002
ERAI_PATH	ERA interim data set (.5°,NetCDF)	1989-2009
ECNC_PATH	ECMWF high-resolution data set (1/4°,NetCDF)	6/11/2006– today
TRIP_PATH	River pathway from Oki and Sund on a 5 km	climatology
WOA2005_PATH	World Ocean Atlas (2005) data set (1°NetCDF)	climatology

Table 3.3: Some of the environment variables set by the file **REGION.src** related to forcing datasets.

seawifs_mon_kpar.sh This routine can be used to set up the water type (Jerlov) based on observations from the SEAWIFS satellite. The use of this data set in hycom runs depends on the setting of the parameter "jerlv0" in **blkdat.input**. Uses routines in the HCYOM **ALL** directory.

3.5.1 Prerequisites for generating forcing fields

To generate the forcing files several environment variables have to be set in the file **REGION.src**. First of all, the variables HYCOM_ALL and MSCPROGS need to be set. They point to the location of the standard HYCOM utility routines, as well as utility routines developed at NERSC. Additionally, several environment variables pointing to the location of data sets are needed, they are: The variables in Table 3.3 need to be changed whenever the location of the data changes or when the system is ported to a new machine.

3.6 Nesting setup (subdirectory nest_nersc)

The nesting in version 2.2 follows closely the approach of standard HYCOM. The contents of the nesting directory will typically look something like this:

```
drwxr-x--- 3 fanf nersc 4096 2011-07-19 11:38 010
drwxr-x--- 3 fanf nersc 4096 2011-07-19 11:38 011
drwxr-x--- 2 fanf nersc 4096 2011-07-19 11:54 bin
```

Where folder 010 contains files related to nesting for experiment 01.0, and folder 011 for experiment 01.1. The folder will contain a folder **outer** if it provides boundary condition, and a folder **inner** if it uses boundary condition. An experiment can be at both time inner and outer, and can provide boundary condition to multiple models.

The **nest_nersc/bin/** subdirectory contains two routines for setting up data files for model nesting. Note that the nestbat routine (**topo/nestbat.sh**) that smooth the topography of the inner model to that of the inner one must already have been run (see Section 3.2.2). The routines in **nest_nersc/bin/** contains setup routines for the two situations where the model provides nesting files to a model, **nest_outer.sh**, and when the model receives nesting conditions from a model, **nest_inner.sh**. Both of these must be run before starting the model. [1]

nest_outer.sh produces necessary files to allow the model to *dump* nesting conditions for another model (referred as inner model). The outer model must know about the grid of the "inner model". The script use 3 arguments as input: the outer experiment number, the path to the region directory of the inner model and the experiment of the inner model. All necessary files for the nesting will be stored in the directory **nest_nersc/$E/outer/** directory. A diagnostic netcdf file is also dump to make sure everything is configured correctly. Finally the script creates/modifies the file **nesting.in** in the experiment directory to indicate where the files will be dump. You must make sure that the targeted folder has writing permission.

nest_inner.sh Run this script before running a model which *receives* nesting conditions. It will set up a relaxation coefficients masks (calles relax_rmu.[a,b]) that will be used for the the baroclinic fields

[1] The need to set up a model before doing outer nesting is new in this version of NERSC-HYCOM.

from the outer model. As for lateral boundary, 20 grid cells is recommended. It will also produce a file **ports.input.nest** that is needed to handle the boundary condition for the barotropic fields. As explained in Section , there are some rules that you need to double check here. Make sure that there is no island, or change in the coast line within these grid cells (try even to do that for a couple more: e.g. 5).

3.7 Tide setup (subdirectory tides_nersc)

In version 2.2 there are several ways of enabling tides in the model. In standard HYCOM tides can be enabled as a body force (mainly relevant for global models), and as a forcing input on the model boundaries. This is done in the **blkdat.input** in the experiment directory. In addition the tidal forcing used in earlier versions of NERSC HYCOM can be used. This approach uses tidal Atlas, which is not used by standard HYCOM.

The routines that are used to set up the NERSC tidal forcing are given in the directory **tides_nersc/bin/**. The routines inside that directory can be used to set up tidal forcing from two different tidal databases.

tides_fes.sh sets up tidal data along model boundary using the FES2004 database.

tides_csr.sh sets up tidal data along model boundary using the CSR database.

Both routines also create diagnostics of phase and amplitude data from he database, interpolated onto model grid points. To access the FES and CSR data sets, the variables CSR_PATH and FES_PATH pointing to the dataset paths need to be set up in the **REGION.src** file.

3.8 Other setup routines (subdirectory other_nersc)

This is the folder where remaining useful script are located. So far it only contains the script curviint.sh.

curviint.sh interpolates the initial state from another model. When initialising a model from climatology, the necessary spin up time is usually of 5–10 years. Starting from a model already in equilibrium allows to reduce the necessary spin up time (approximately 1 year). This approach is typically used in nested configuration where the initial state for the inner model is generated from the outer one. The program assumes that the model that it interpolates from includes the local domain, that both grid are orthogonal curvilinear and that the vertical structure are the same (same number of layers and reference densities).

Chapter 4

Model Code

The NERSC-HYCOM code is based on the standard HYCOM code, but there are some additional features. The new features are mainly in the domain of ice modelling (sea-ice dynamics and thermodynamics) and data assimilation (for running perturbed ensemble simulation). There are also changes due to diagnostics (weekly and daily averages) All the new routines from NERSC are located in a separate subfolder (**nersc**) and when the original file from HYCOM is changed the change are easily identified using the CPP FLAG **NERSC_VERSION**.Work is done to converge as much as possible with the standard way of doing things in HYCOM. For example, the barotropic/baroclinic nesting routines now rely on the standard HYCOM approach, with only input/output and addition of tides being different from the standard version. Another example is the conformal mapping routines which in this version is completely decoupled from the HYCOM model code. The conformal mapping routines are only used by model setup routines in Section 3.

NB: For a experienced user the checklist in Section A may be used. It gives step-by-step instructions on how to set up the model.

4.1 Differences from standard HYCOM

The discrepancies from the standard HYCOM are now detailed. Most of the option will be activated by CPP flags when you are compiling the code, also some are specified in input files: infile.in, infile2.in, infile.evp, infile.icestate. The changes are group in different classes: Input/Output (restart name convention and run limits), Diagnostics (daily, weekly); tides (Atlas choices, add currents); nesting; ice dynamics; thermodynamics, and atmospheric forcing.

4.1.1 Input/Output

In NERSC-HYCOM, the output times can be determined directly in the input file called **infile.in**. This file also determines the start and end times of a model integration, as well as some other parameters for the model runs. The FORTRAN routine for setting up the integration times and diagnostic outputs is **src_2.2.12/nersc/m_limits.F90**. This file will set up the diagnostic times using types defined in the module **src_2.2.12/nersc/mod_diagnostics.F90**. The file names convention in NERSC-HYCOM is also different from the standard HYCOM. In standard HYCOM restart files are simply called **restart.in** or **restart.out**. In NERSC-HYCOM the time info is a part of the restart file name, as well as a three-letter ID called "rungen" - given on the second line of the file **infile.in**. A typical file name for a restart file in NERSC-HYCOM is

IDBrestart1988_050_00.[ab]

where "IDB" is the three-letter ID, and after "restart" follows the year, day of year (a.k.a ordinal date) and hour. For the diagnostics files, or "archv" files, the name is the same as in standard HYCOM, but the rungen is used as a file name prefix. Note that in NERSC archv are very seldom used because it exists other type of diagnostic files (daily average, weekly average).

The choice of different file names means that there are slight changes in the standard HYCOM FORTRAN files **src_2.2.12/restart.F** and **src_2.2.12/archv.F**. Also, when it comes to restart files, HYCOM has traditionally had several restart files for different model components, such as biological, ice and ice dynamics models. In version 2.2 many of these restart files are removed, and the fields are in stead dumped

to the HYCOM restart file, leaving a single restart file. This is done by setting the CPP flag "SINGLE_RESTART". Some restart files, such as the ice model files, are still saved to separate restart files. This can be switched off, however, by setting the CPP flag "SINGLE_RESTART_ONLY".

4.1.2 Diagnostics

In addition to the standard HYCOM diagnostics, it is possible to create extra diagnostics from model runs. Two such extra diagnostics are daily and weekly averages, which must be enabled during compilation with CPP flags, and in addition they must be enabled in the file **infile.in** when running the model. The reason for using CPP flags for these two options is that they require 3D variables to store the averaged data and will requires a little bit more memory (this can be switched off).

Daily averages are processed by the module **src_2.2.12/nersc/mod_daily_average.F90**. To use daily averages they must be set up by setting the CPP flag "DAILY_AVERAGE", and they must be activated at run-time in **infile.in**. The model fields are average at every model time step. Weekly averages are processed by the module **src_2.2.12/nersc/mod_average.F90**. To use weekly averages they must set up by setting the CPP flag "WEEKLY_AVERAGE", and they must be activated in **infile.in**. You need to provide in the infile.in how often you use the model output for averaging (typically 2 hours).

It is also possible to diagnose grid point time series sampled every hour. This is done by the FORTRAN module **src_2.2.12/nersc/mod_gridp.F90**. This is activated at run-time in **infile.in**.

4.1.3 Nesting

The nesting in NERSC-HYCOM is now very similar to that of the standard HYCOM. Because the nesting in standard HYCOM was introduced at a later stage than in NERSC-HYCOM there were several duplicate routines for doing the nesting. This included the application of nesting to baroclinic and barotropic variables, which had to be put into the standard code at various places. In the new version of NERSC-HYCOM the procedure of standard HYCOM is followed, the exception being the input and output of nesting files. The input and output of nesting fields still follows conventions particular to NERSC-HYCOM. However, with version 2.2 it should be possible to use the nesting input/output approach of standard HYCOM as well.

The FORTRAN module **src_2.2.12/nersc/mod_nesting.F90** is used to initialize, write and read nesting conditions. The actual calling of NERSC nesting routines is done inside the main hycom FORTRAN files **src_2.2.12/mod_hycom.F90**, where it is used when saving nesting conditions, and inside the routine **src_2.2.12/forfun.F**, where it is used to read nesting conditions. The routine which reads nesting conditions, **nestcondread**, is a drop-in replacement for the equivalent routines in standard HYCOM **forfun.f**: **rdnest_in** and **rdbaro_in**.

To let a model save nesting conditions to be used in another model, it must be activated in **infile.in**. This is done on line 11 where you specify the outer nesting flag, and the file saving interval of nesting conditions (in hours). For a model to read nesting conditions line 12 of **infile.in** must be edited, where the inner nesting flag and the nesting read interval is specified. In addition inner nesting must be enabled at compile time by setting the CPP flag "INNER_NESTING". If your model includes ICE, your need also to define the flag ICE_NEST, to save ice properties as well.

4.1.4 Tides

In version 2.2, tidal forcing has been introduced in standard HYCOM. In NERSC-HYCOM the tidal forcing has been present for a much longer time. The tidal forcing in NERSC-HYCOM is based on the use of tidal atlases, where the sea-surface elevation is specified on the boundary using prescribed amplitudes, periods and phases shifts. The approach in standard HYCOM is somewhat different, where (AFAIK) tidal atlases are not used. In standard HYCOM it is the equilibrium tide which is used, excluding the effects associated with bathymetry and basin configuration.

The tides are handled by the FORTRAN module **src_2.2.12/nersc/mod_tides_nersc.F90**. This module is fairly self-contained. It is called from the main hycom routine **src_2.2.12/mod_hycom.F** for initialization and for updating the tidal constituents at regular intervals. It is also called from the FORTRAN file **src_2.2.12/latbdy.F**, where it is used to supply the correct boundary conditions during nesting. Presently inner nesting must be enabled in order to use the NERSC tidal module. Some restrictions are set on the use of the NERSC tidal flag in NERSC-HYCOM. The most obvious is that tidal forcing using the NERSC tidal module and the standard HYCOM tidal module can not both be active at the same time.

4.1.5 Ice dynamics: EVP, MIZ and ICESTATE

Ice modelling is one of the main topic of interest and area of development of the NERSC-HYCOM. The code is very different from the standard HYCOM in this regard. Recently, effort are made in COAPS to couple the standard HYCOM with CICE model which may allows us to converge to the same distribution in the future. Ice modelling at NERSC are fairly comparable to CICE model, but the ice model at nersc is coded on a Arakawa C-grid instead of B-grid as in CICE in order to reach a better agreements with HYCOM. NERSC-HYCOM gives the possibility to use a one thickness ice model (CPP flag ICE) or multi category ice model (CPP flag ICESTATE). You also get the possibility to use two different ice-rheologies: EVP and MIZ. In Elasticous Visous Plastic (EVP) the ice rheology will depend on the shear in x and y, and you will transit from a viscous rehology (if the stresses are within the ellipse) to a plastic rehology (if the stress are out of the ellipse). Such behaviour is well suited in pack ice and allows for ridging, crack, ice bridge. However, when the scale of the ice gets smaller (at the ice edge), this behaviour is no longer suited, and the rheology becomes collisional (Marginal ice zone rheology). It is quite difficult to identify where the rheology should transit from EVP to MIZ. Ideologically, the difference represent a different type of ice. At the edge, the concentration is lower, and the ice is forming panckake. Two approaches are proposed in the NERSC-HYCOM code. A first one is based on ice concentration (CPP flag EVP+MIZ) while a second is based of the floe size that are break by the waves (CPP flag EVP+MIZ+WAVES). The latter is currently in a development stage. By default now, the advection of ice properties is calculated using a 3rd order WENO scheme, with a 2nd order Runge-Kutta time discretisation. All the code related to the ice modelling are located in **src_2.2.12/nersc/EVP** for EVP and MIZ and in ICESTATE for the multi-category ice model.

4.1.6 Thermodynamic forcing

The thermodynamic in NERSC-HYCOM over open water, ice covered and snow covered ice is based on the formulation from Drange and Simonsen 1996 with a correction of heat fluxes for subgrid scale ice thickness heterogeneities following Fichefet and Morales Maqueda [1997]. The surface fluxes are forced with a bulk formula parametrisation [Kara, 2000]. The short wave radiation are estimated theoretically, based on the cloud coverage. The latter is by default updated every day, but you have the possibility to update the field every 3 hours (CPP flag **DIURNAL**). There is 2 ways to define the momentum flux: the one from Large and Pond [1981] that is used by default, or the one from Kara et al. 2002 that use a more advance formula with respect to the wind speed and that accounts for discrepancies between the model SST and air temperature (CPP flag **KARA2002**).

4.1.7 Atmospheric forcing

The atmospheric forcing in NERSC-HYCOM is based on that of the standard HYCOM version. However, since the thermodynamical forcing is different, there are some extra fields used in NERSC-HYCOM. These fields include winds (standard HYCOM uses only atmospheric stress), relative humidity, clouds, and sea-level pressure. To get these fields some changes (or rather, additions) have been introduced in the file **src_2.2.12/forfun.F**. This file is based on standard HYCOM, but the reading of the new fields has been introduced. The modified **forfun.F** uses parameters from the FORTRAN module **src_2.2.12/nersc/mod_forcing_ne** and the added atmospheric fields are also read into variables defined in that file.

It is possible to force the model in a manner similar to standard HYCOM, where forcing input is read from the "ab"-format. This requires that forcing fields are preprocessed using NERSC utility routine described in Section 6.4.36. Otherwise, you get the possibility to read synoptic forcing directly from few selected data sets :ERA40 dataset (NetCDF files), the NCEPR data set (NetCDF files, aka "ncep"), ECMWF T799 (NetCDF, aka "ecnc"), ECMWF T159 (binary files, aka "ecmwf"), ECMWF T399 (binary files, aka "metno"), and the ERA interim reanalysis (ECMWF T399 (netcdf files, aka "era-i"). In order to use this option, the CPP flag "FORCING_INLINE" must be set when compiling the code. For further information on setting up the forcing are described in Section 3.2.

4.1.8 Random forcing

The random forcing module is used to modify the atmospheric forcing used in NERSC-HYCOM. By using the routines of this module, fields with a prescribed variance and correlation in time and space will be added to the original forcing fields. The random forcing features are mainly used when running a model ensemble,

and its purpose is to produce a spread in the ensemble, which is important when doing data assimilation using the EnKF.

The random forcing is taken care of by the FORTRAN module **src_2.2.12/nersc/mod_random_forcing.F90**. the forcing fields are assumed to be red noise simulated by a spectral method as described in (**?**). The perturbations are computed in a Fourier space with a decorrelation time-scale and horizontal decorrelation length scale. Variables to be perturbed, time and space decorrelation time scale are provided in the input file (**infile2.in** as described in Section 5.1.4. It is possible to derive wind field perturbations geostrophically from the SLP perturbations, their intensity being inversely proportional to the value of the Coriolis parameter. This ensure some kind of consistency in the perturbed field. The standard deviation set in the input file correspond to 40°N. There is a smooth transition to the Equator, where winds get aligned with the gradients of SLP perturbations. In this module are all routines related to the random forcing. In addition the value of the random forcing are stored in the restart files to avoid discontinuity in the perturbation. This is handled by the restart routines in **src_2.2.??/nersc/mod_restart.F90**. There is also a possibility to perturb the ice dynamics by perturbing the squared parameter e^2 in the EVP rheology. This parameter represents the ratio between the minor and the major axis of the elliptic yield curve, which partly controls the transition between the viscous and plastic flows for a given stress. In other words, it represents the shear to compression strength ratio. If the CPP flag PERTURB_ICE is activated, this parameter is chosen randomly from a Gamma distribution (k=1 and σ =5). To use this flag you need to have the DRANDGAMMA library available present in the acml library on hexagon (module load acml). **src_2.2.??/nersc/EVP/evp_init.F**.

4.2 Code setup

This section deals with the setup of the model code, meaning the setting of CPP compilation flags, the code itself , configuration files and Build directory, and the preprocessing of model dimensions.

4.2.1 Setup of directories

To compile the HYCOM model three directories are needed: **src_2.2.12/**, **config/**, and a Build directory that corresponds to a given experiment, (the name will be **/Build_V2.2.12_X01.0/** for expt 01.0). Template are available in the svn copy:

~/hycom/HYCOM_2.2.12/CodeOnly/

However, the creation of these files should be automatic if the script **newRegion.sh** and **newExp.sh** have been used (See Section 3.3). These should be handle automatically by the script. Beware that (**src_2.2.12/**) and config directory (**config/**) are linked to the svn copy of the code, so that the model makes always use of the most recent code. The Build directory is copied physically as it contains files that details the model parametrisation and are thus dependent on the experiment.

4.2.2 Setup of model dimensions

When setting up the model code, the grid dimensions as well as MPI setup needs to be specified. This is provided inside the file **dimensions_nersc.h** inside the Build directory. This file is created automatically by a shell script. To do so, go into the Build directory corresponding to your experiment, and run the script **setuppatch.sh**. This script is called with the number of MPI domains as its only argument, for example:

setuppatch.sh 51

The command above will look for "patch" files which have a total of 51 MPI. The "patch" files created in Section 3.3.4 and placed in the directory **topo/partit/** relative to the top-level region directory. The names of the patch files are related to the topography version of the experiment number (set in exptXX.X EXPT.src) and the name of the region. A typical "patch" files names would be:

topo/partit/depth_FRMa0.05_01.0051

where in this example "FRMa0.05" is the region name, "01" is the topography version, and the last number, "0051", is the total number of MPI tiles for this patch file. The script will return an error message if the file is not found.

The file created **dimensions_nersc.h** looks like this:

```
integer, parameter :: itdm=400
integer, parameter :: jtdm=320
integer, parameter :: kdm=28
integer, parameter :: idm=50
integer, parameter :: jdm=46
integer, parameter :: iqr=8
integer, parameter :: jqr=7
```

where *itdm*, *jtdm* and *kdm* are the 3D dimensions of the model domain retrieve automatically from the blkdat.input. *idm* and *jdm* are the maximum horizontal dimensions of a tile (MPI sub-domain), and *iqr* and *jqr* are the maxim number of tiles in the two horizontal grid dimensions.

4.2.3 Setup of Makefile configuration files

To compile the model, it is necessary to specify the correct config file to include. This file specifies the compiler and compiler flags to use, as well as some of the CPP flags needed for different architectures. The directory **config/** under the top-level region directory contains configuration files for various architectures. Their names are typically linked to architecture and parallelization types. As an example, the config file **config/xt4_mpi** contains makefile macros needed when compiling HYCOM for the Cray XT4, and using MPI parallelization (hexagon,Bergen). The model is also tested on other machine: IBM AIX 5L (NJORD, Trondheim) **config/sp5_mpi**; intelCHPC (Cape town) **config/intelCHPC_mpi**.

The files in the **config/** directory have been obtained from the standard HYCOM distribution, and can be used as starting points for creating new config files. In most cases, however, it should be possible to use these without modification when compiling the model.

To set up the Makefile to use a specific configuration file, the file **flags** in the build directory needs to be modified. At the very top of this file two variables are defined, which is used to include the correct configuration file:

```
ARCH=xt4
TYPE=mpi
```

The variable ARCH denotes the architecture to compile for and the variable TYPE denotes the parallelization type to use. This will be used to read the correct config file into the makefile named **Makefile**. With the variable definitions used above, the makefile will look for the configuration file **config/xt4_mpi**.

In addition it is sometimes necessary to include libraries when compiling the model. This can be set in the configuration files, but it can be more convenient to set them in the file **flags**, where the variables FCFFLAGSNERSC and LIBS should be set according to the location of the libraries you need to include. As an example, consider including the NetCDF libraries, which can usually be included in a manner similar to this

```
LIBS := $(LIBS) -L/usr/local/lib/ -lnetcdf
FCFFLAGSNERSC := $(FCFFLAGSNERSC) -I/usr/local/include/
```

this assumes that the NetCDF include files are installed in **/usr/local/include/**, and that the libraries are installed in **/usr/local/lib/**. The location of libraries differ from machine to machine, so this must usually be modified when porting the code.

4.2.4 Setup of CPP flags

The CPP FLAGS are an easy way to chose some part of the code. These choice are made in the file named **flags** in the build directory. Here various CPP options from the standard HYCOM distribution can be set, as well as CPP options particular to NERSC-HYCOM. Table 4.1 lists the various NERSC-related CPP options used when compiling the model.

These flags need to be set in the file **flag** inside the build directory. Look for the variable CPPFLAGSNERSC and edit it to include the features you want to use in the model, the following example;

CPPFLAGSNERSC := $(CPPFLAGSNERSC) -DNERSC_VERSION -DNEST_INNER -DICE -DEVP

will switch on inner nesting, the standard ice model and the evp sea-ice dynamics model.NB: Do not switch off NERSC_VERSION unless you know what you are doing, the compilation will likely fail.

4.2.5 Compilation

Assuming everything is set up, the model is now ready for compilation. To compile, simply type

```
make
```

Note that it is no longer necessary to do "make new", since the source files and the dependencies needed to compile the model are set up in the the config files and the include files **flags** and **dependencies**. Note that you need to clean the Build everytime you are editing the flags file or the make file, but not when you edit the source code. To clean up the directory, type:

```
make clean
```

If the compilation fails, there can be several reasons. Typical culprits are

- Missing **dimensions_nersc.h**, which must be set up before compiling, see Section 4.2.2.
- Missing config files in the **config/** directory, make sure that the ARCH and TYPE variables are set up properly in the file **flags** in the build directory.
- Invalid config files in the **config/** directory. When porting the code to a new machine, it is sometimes necessary to create a new config file. Use one of the existing config files in the directory **config/** as a starting point.
- Missing libraries and/or include files, make sure they are set up properly as described in Section 4.2.
- Missing source files, make sure they are set in the file **flags**. They should be, but you never know. If HYCOM is modified, and new FORTRAN files are created, you will have to add them to the file listing in **flags**.
- Missing dependencies. When the model is compiled and FORTRAN90 modules are used, dependencies need to be set up. These are normally set in the file **dependencies**. If new FORTRAN90 modules are used with the NERSC-HYCOM code, they may need to be added in this file.

Variable	Description
NERSC_VERSION*	Use NERSC-HYCOM code and any of the flag below (Must be defined).
MODEL DYNAMICS	
DIURNAL*	Update the clouds fields every 3 hours instead of daily for shortwave flux
FORCING_INLINE*	Read forcing data directly from raw datasets instead of standard HYCOM preprocessed files
MATS_RELAX*	turn off relaxation of surface salinity when departure from climatology is larger than .5 psu. This is deprecated in the version 2.2.37 of HYCOM, and you should use negative sssflg in blkdat to activate it, See Section 5.1.1
SEASONAL_PORT	Allow seasonal variability in the barotropic inflow (input file needed)
INTERANNUAL_PORT	Allow inter-annual variability in the barotropic inflow (input files needed)
NEST_INNER*	Applying NERSC-HYCOM nesting conditions at the boundary of the model.
NEST_OUTER*	Saving NERSC-HYCOM nesting conditions
SINGLE_RESTART*	Put all models variables in .(a,b) files still dump old ICE and EVP files
SINGLE_RESTART_ONLY*	Stop dumping ICE.uf EVP.uf files
ICE MODELLING	
ICE*	NERSC-HYCOM standard heat flux formulation. (req. EVP)
EVP*	Standard EVP sea-ice model (req. ICE)
MIZ	Transition to collisional rheology (Marginal ice zone) in low concentration ice (req. EVP, ICE)
WAVES**	Transition to MIZ estimated from floe size (calculated from wave field, req. EVP, MIZ)
ICE_NEST*	Activates sea-ice nesting when running (req. NEST_INNER).
ICESTATE	Advanced multi-category sea ice model
TEST_ICE_AGE	Compute ice age as seen by satellite (funct(compression,age),req. EVP,ICE)
ICEAGE	Compute ice age
ALBSNW_EVOL	Albedo evolution based on Douville et al (95), decrease with ageing (req. EVP,ICE)
SSNOWD_ICE**	use a snow model (req. EVP,ICE)
SSNOWD**	use a snow model (req. ICESTATE)
ECOSYSTEM MODEL	
NOR05	Switch on NORWECOM ecosystem model
ZOOPL	Add zooplankton to NORWECOM (req. NOR05)
DETP	Add phosphorous detritus to NORWECOM (req. NOR05)
NEST_BIO	Nesting for ecosystem model NORWECOM (req. NOR05)
CALANUS	Individual based model for C. finmarchicus
ASSIMILATION	
EVP_DRIFT_ASSIMILATION	Makes HYCOM dump daily averages of sea ice drift from an ensemble
TRACK_ASSIM	Makes HYCOM dump daily averages of sea surface height from an ensemble
PARAM_EST	Add bias estimate values for SSH, SST and sea ice strength
PERTURB_ICE	Perturb the ratio of the ellipse in the EVP to get more spread in the ice ensemble
DIAGNOSTICS	
WEEKLY_AVERAGE	Used to dump on weekly averages of the model
DAILY_AVERAGE	Used to dump on daily averages of the model
DAILY_ENSVAR	Dump the squared of ssh, fice, hice to compute the full variance (diurnal/ensemble)
ICE_DYN_DIAG	stored diagnostic variables from EVP in daily average (stress,pressure, tensor)

Table 4.1: CPP Flags used for setting up NERSC-HYCOM. The term "req." indicates that some conjugate flag are required. * indicates that the flag is recommend by default for standard implementation.** indicates that the flag is not fully tested yet.

Chapter 5

Final model setup and running the model

This chapter describes how to run the model, assuming that it has been compiled, and that the data files necessary to run the model have already been generated. It also assumes that the experiment subdirectory has been set up properly.

NB: For a experienced user the checklist in Section A may be used. It gives step-by-step instructions on how to set up the model.

5.1 Overview of a model run

Running a model is different from the procedure used in earlier versions of NERSC-HYCOM, where every input file was put into the model run directory manually. With the new procedure everything is done automatically but it relies on the region directory hierarchy, and that all necessary input files are present. Everything is now happening for a given experiment in the region dir. The experiment directory will typically look like that:

```
-rwxr-xr-x 1 fanf fanf  11876 2011-07-07 10:05 blkdat.input
drwxr-xr-x 3 fanf fanf   4096 2009-03-12 11:47 data
-rwxr-xr-x 1 fanf fanf   1216 2009-02-16 14:37 EXPT.src
-rwxr-xr-x 1 fanf fanf    667 2009-02-16 14:37 infile2.in
-rwxr-xr-x 1 fanf fanf    118 2009-02-16 14:37 infile.evp
-rwxr-xr-x 1 fanf fanf    633 2009-02-16 14:37 infile.icestate
-rwxr-xr-x 1 fanf fanf   1296 2009-02-16 14:37 infile.in
drwxr-xr-x 3 fanf fanf   4096 2009-03-12 11:47 log
-rwxr-xr-x 1 fanf fanf   1941 2009-02-16 14:37 pbsjob.sh
-rwxr-xr-x 1 fanf fanf    452 2009-02-16 14:37 ports.input
-rwxr-xr-x 1 fanf fanf   1309 2009-02-16 14:37 postprocess.sh
-rwxr-xr-x 1 fanf nersc 21164 2010-09-03 12:09 preprocess.sh
-rwxr-xr-x 1 fanf fanf   2240 2009-02-16 14:37 README
drwxr-x--- 2 fanf nersc  4096 2011-07-29 15:15 SCRATCH
drwxr-xr-x 5 fanf fanf   4096 2009-03-12 11:47 subprogs
```

The files **EXPT.src**, **blkdat.input**, **ports.input**, contains information about the model configurations. These have already been described set in Section 3.3, and only minor changes may be applied. The files **infile.in**, **infile.evp**, and **infile2.in** contains information about: when to start your model, if you are going to store nesting, use tides, dump weekly/daily average. The file **blkdat.input** contains informations about the vertical structure and model parametrisation (viscosity, turbulence model, ...). There is also 4 folders, **data**, **SCRATCH**, **log**, and **subprogs**. The latter contains useful programs in python and matlab script that are independent of the model. The data directory, contains the model outputs gathered during the runs. The log directory, contains the logfiles from the last run (possible to change the main script so that it dump a name that depends on the job number). The SCRATCH directory is a temporary directory where all necessary file are copied before the model is run. Finally, there are three shell scripts **pbsjob.sh**, **preprocess.sh**, and **postprocess.sh**. The script pbsjob.sh is the main job script that need to be submitted to the machine. It is machine dependent and should contains information about the time of the job, account number, number of cpu, memory usage, etc. You will only need to call this one as it calls the two others.

preprocess.sh copies all necessary files to the SCRATCH directory and returns an error message if some files are missing. **postprocess.sh** is called once the job is completed, and move all model output files to the data dir.

5.1.1 blkdat.input

This is the standard configuration file used when running HYCOM, and is identical to that of standard HYCOM. Here we describe only the model parameter that are independent from the model vertical structure. Only the one that we have expertise in are described, otherwise you can refer to the standard HYCOM doc.

- iniflg and yrflag are useful when restarting model from climatology (See Section 5.2.3). Otherwise use yrflag=3.
- jerlv0 allows a spatially varying light penetration that depends on KPAR climatology (recommended by default; i.e.=1).
- sshflg to allow getting the steric height contribution of the SSH in a an additional variable.
- All dsurfq, diagfq, proffq, tilefq, meanfq, rstrfq,cplifq are not used in NERSC VERSION.
- nestfq and bnstfq are used for inner nested model. Typical values used a NERSC is 0.25 for both.
- baclin and batrop are the baroclinic and barotropic time step of your model, that MUST be tuned to satisfy the CFL at your model resolution.This is very important to do, because you may speed up considerably your model by tuning these parameter without affecting the accuracy of you results. The two values cannot be chosen independently and a matlab script **subprogs/micomstep.m**. The script uses two arguments (range of baroclinic and barotropic time steps) and returns all the possibilities.
- incflg incstp and incup are used for incremental update. These are use when data assimilation does not maintain dynamical balance, but introduce a time shift in the update. Not necessary when using data assimilation method used at NERSC.
- wbaro and btrlfr are related to the time discretisation. Leapfrog is recommended, but is is unconditionally unstable and Asselin filter is used for stability. (wbaro=0.125 is recommended).
- hybmap represents the vertical interpolation within the mix layer. the WENO like PPM is highly recommended.
- hybflg and advflg decide what variable to advect, The two should be equal. If you use ice, use 0, otherwise the option 1 is better.
- advflg select wich advection scheme to be used. MPDATA is deprecated because it is positive definite but no monotonous (doesn't keep the initial range of the advected quantity). The FCT4 is best but often unstable, so standard is FCT2.
- momtyp and facdf4 define which momentum advection type you are using. momtyp=4 is best, but is a little unstable in places of strong flows. If momtum=4 is chosen, facdf4 must be given a value (from 1/32 for coarse model to 1/200 to high resolution model), veldf4 and veldf2 must be 0 and visco4 must be set to for example 0.05 and visco2 to 0. If momtyp is set to 2, you first need to decide if you prefer to use laplacian viscosity or biharmonic viscosity. The former is more realistic and smoother, and the later gives usually better results. Once you are satisfy with your viscosity, you need to tune the velocity diffusion and the thickness diffusion to reduce the noise. Noise in the velocity is often appearing at the surface, while noise in the dp typically appears bellow the mix layer depths. Here, biharmonic diffusion is recommended (i.e. thkdf4 and veldf4). By experience the velocity diffusion is not problematic, but the thickness diffusion can be very destructive of the stratification and its usage should be limited. It is possible to set the values as negative, then HYCOM will use a spatially varying values that it will read from a file with similar name (Tools for producing these files are describe in MSCPROGS/THKDF4)

- thkmls is used when sssflg is activated- when surface salinity is relaxed. The actual relaxation depends on the effective mixed layer depth. The thkml variable sets the reference mixed layer depth for a 30-day e-folding time. The actual e-folding time is 30*MLD/thkml (shorter means a stronger relax). If a negative value is given, HYCOM will look for a file sssrmx.[a,b] that correspond to a threshold above which the relaxation is not proceeded. The reason is that relaxation can have a detrimental impact on some regions particularly where strong fronts occur and/or they are misplaced (e.g., Gulf Stream). In such places the water mass distribution is bimodal, and the relaxation towards an average estimate reduces the sharpness of fronts. Typical value for the threshold is 0.5 psu (Mats Bentsen).

- mlflag defines which turbulence model you are using for the mixed layer depths. KPP is usually the most recommended as it is well tuned and tested. However it requires a lot of tuning and lot of different option (e.g. bblkpp to include a bottom boundary layer). There is a document made by SHOM that describe KPP in detail. GISS is an improved version of Mellor-Yamadaand thus better suited for near coastal area. By experience, it is the one that provides the best result in TOPAZ, but as a different mixing is computed for each point it can be noisy. Another advantage of the scheme is that it requires very little tuning.

- tsofrq, tofset, and sofset can be used if you observed a drift in you model in T and S. It is possible to adjust for that by adding a value to the mode (value corresponds to the drift/century).

- lbflag indicates how you treat your open boundary condition. A value of 2 is used with nesting. The option 3 is used for flather boundary condition (not tested yet). Value 0 and 1 assume that the boundary is a solid wall. The value 1 is used when you want to add a barotropic port in your model (ports.input is necessary, see Section 3.6). This version of HYCOM seems to have problems when it comes to total volume conservation. Therefore, even if you don't want a port, it is recommended to set lbflg=1 and set the nb of port to 0 in the ports.input, then you will benefit from the fix.

- wndflg. Unfortunately there is still a mystery for me. We should use wndflg=3 because the atmospheric forcing are on a p-grid, but this makes more noise (checker boarding). Therefore a value of 1 seems to perform best.

- difsmo is an engineering (efficient) fix to remove checker boarding from the model surface. The value should be equal to the number of layer you want to remove it.

- flxflg must be set to 99 when using the NERSC version

- relax to activate nudging from relaxation at the boundary.

- priver and epmass. The former activate the river forcing and the latter provides an additional barotropic signal to it.

The other parameter have not be tested and are based on the blkdat.input examples from Alan Wallcraft and Zulema Garaffo.

5.1.2 EXPT.src

This file has also been covered in Section 3.3. For the purpose of running the model, two definitions are important in this file; the definition of the "scratch" directory, where the model will run, and the definition of the "data" directory, where model results are stored. Both of these should be set to appropriate directories by the user. The following is an example of their definition.

```
export D=$P/data                                  # Where data ends up
export S=/work/$USER/hycom/$R/expt_$X/SCRATCH   # Scratch area
```

The directories will of course have to be accessible in order for the preprocessing and postprocessing scripts to work. If the directories do not exist, a new one will be created.

5.1.3 infile.in

This file is particular to NERSC-HYCOM. This file determines the start and end times of a model integration, sets the diagnostic times and diagnostics type as well as some flags needed to run NERSC-HYCOM. Note that this file assumes that the date are given in day in year (days since the beginning of the year). An example of the contents of this file with line numbers added is given here(line numbers added):

```
1   3.1          # version of inputfile (limits.dat)           DO NOT CHANGE
2   IDB          # (rungen)   Version number of run
3   1988         # Reference year for simulation  (365. for spinup)
4      5 00      # (nday1)    First day of integration NOTE FORMAT F9.2
5     90 00      # (nday2)    Last day of integration  NOTE FORMAT F9.2
6   era40 era40  # forcing option, month, ecmwf, ncepr, ecmo, ecnc
7    200.0       # temperature relaxation time scale (F in blkdat.input)
8    200.0       # salinity    relaxation time scale
9   F  2         # laverage n Accumulate monthly averages every n hours
10  F  T         # Switch on and accumulate (T) or overwrite (F) daily averages
11  F  6         # lnesto, nestdto - saves nesting bnd cond at nestdto intervals
12  T  6         # lnesti, nestdti - read and apply nesting bnd cond at nestdto intervals
13  T FES F F    # Tides (true,  CSR/FES, apply currents)
14  F            # lgridp     Activate storage of gridpoint information
15  # Days     Assi  Diagno.  Restart           (1992 for first synoptic experiment)
16       5.00  F     T        T
17      60.00  F     T        T
18      90.00  F     T        T
```

The above will set the model up to start at day 5 in 1988 and stop at day 90 in 1988. Restart and diagnostic files will also be created for day 60. In you want to extend your run beyond year 1988, just set nday2 to a value larger than 365. The Final restart file will be dump with the correct year (here 1989) with the corresponding year. In HYCOM the start and end times of an integration are also set in a file called "limits". This file is used in NERSC-HYCOM v2.2 as well, but it is calculated from the file **infile.in** by the preprocessing script **preprocess.sh**.

The version of the input file is given in the first line, NERSC-HYCOM will check that this is correct before model integration starts. The second line is a three-letter ID commonly referred to as "rungen". When the model creates diagnostic and restart files, the rungen will be placed at the start of the file name.

The third, fourth and fifth line is used to set integration ranges. A reference year is given, as well as start and end times of the model integration in days and hours. Both the start and end days are relative to the first day of the reference year.

The sixth line is used to choose the forcing data set to use. There are two flags, the first is the synoptic forcing data set to use, the second is the climatology data set to use. If the model is run with climatological forcing, the synoptic forcing flag should be set to "month". If the forcing is pre-generated, the synoptic forcing flag should be set to "prep". Forcing can be pre-generated by the script **nersc_synotpic.sh** in the **force/bin/** directory. Table 5.1 shows the possible options for the synoptic and climate forcing flags.

The generation of forcing is described in more detail in Section 3.5. The 7^{th} and 8^{th} line of **infile.in** are no longer used as the relaxation time scale is set by 1/30 days by default in HYCOM (TODO remove these lines). The 9th line of **infile.in** switches on (T) or off (F) the weekly average option. The second item on line 9 denotes how often the (in hours) the model should be sampled to produce the weekly average. Remember that to use weekly averages they must be set up at compilation by setting the CPP flag "WEEKLY_AVERAGE", as described in Section 4.2.4. Weekly averages will be dumped to files with names of the type **XXXAVE_YYYY_MM_WW.[ab]**. The file name contains year, month and week of the start of the model run. Not that "month" and "week" is not real, they should be considered as a way of splitting the year into 48 segments.It implies that in some months some weeks will contains 8 days and some other 7.

Line 10^{th} of **infile.in** is used to activate daily averages. The first value switches them on and off, while the second line will switch on accumulation of daily averages. This "accumulation flag" is mainly

Synoptic flag	Description
month	Used when running NERSC-HYCOM with climatology option
prep	Used when running NERSC-HYCOM with a pre-generated data set (set up with script **nersc_synoptic.sh**).
era40	Use the ERA40 data set
era-i	Use the ERA interim data set
ncepr	Use the NCEP data set
ecmwf	Use the ECMWF T159 operational data set
metno	Use the ECMWF T399 operational data set
ecnc	Use the ECMWF T799 operational data set

Climatology flag	Description
era40	Use ERA40-derived climatology
ncepr	Use NCEP-derived climatology
old	Use "old" climatology (deprecated)

Table 5.1: Synoptic and climatology forcing flags used with NERSC-HYCOM.

used for running ensembles - if it is switched on, the daily averages of the different ensemble members will be accumulated into one daily-ensemble average. For normal single-member runs, the accumulation flag should be set to false. The daily averages will be dumped to files with names of the type **XXXDAILY_YYYY_DDD_YYYY_DDD.[ab]**. The file name contains year and ordinal day of the start of the model run, as well as the year and ordinal day of the daily average. To use daily averages they must be set up at compilation by setting the CPP flag "DAILY_AVERAGE", as described in Section 4.2.4.

Line 11 and 12^{th} of **infile.in** us ised to activate "outer" and "inner" nesting. The logical value (T or F) is used to activate/deactivate the nesting, while the second value, an integer, denotes how often to save nesting (for the outer model) or how often to read nesting conditions (for the inner model). This nesting interval value is given in hours, and the model which saves nesting conditions must have a nesting interval which matches the model which actually receives them. To use inner and/or outer nesting requires preprocessing, described in Section 3.6.

The 13^{th} line is used to switch on tidal forcing with the NERSC-HYCOM approach that is based on tidal atlases. There are four values on this line - the first logical value is to switch onoff tidal forcing. The second value denotes the tidal atlas to use (FESCSR). The third logical value denotes if tidal atlas currents should be used (or if just tidal elevations are used). The last value is a logical which indicates "slowstart", where the tidal forcing will be slowly ramped up from zero to the full tidal elevation values. This is usually switched off, but can be used when tidal forcing is switched on for the first time during a model integration. Normally it should be set to false. To use tidal forcing requires preprocessing, described in Section 3.7.

The 14^{th} line is used to diagnose grid point time series sampled every hour. To do this you need to set the flag to T. You will also need to specify the grid points to process using the file **infile_gp.in**, described in Section 5.1.6.

From line 16-17^{th} you need to specify the date to which there will be input/output of restart or archive files (weekly or daily are automatically dumped). Currently the archive file dump has been switched off because they are not used in NERSC. The first column corresponds to the day in year (i.e starting from 000 with no upper limit); the second and the last should be F, T, T. There are some rules to follow when making the list of dump. There must be a diagnostic time equal to the first and last integration times specified on lines 4 and 5. The list of diagnostic time must be strictly increasing.

5.1.4 infile2.in

This file is used to set up the random forcing used when running a model ensemble. It is typically used for assimilation purposes. Its contents are

```
1.2 'fversn' = version of inputfile (infile2.in)
 0   'randf ' = 1=random forcing, 0= no random forcing
11   'seed  ' = Random forcing seed
```

```
 10.0    'vslp  ' = Variance in slp
  9.e-4  'vtaux ' = Variance in tau_x
  9.e-4  'vtauy ' = Variance in tau_y
  2.5    'vwspd ' = Maximum of the perturbed windspeed.
  4.e-2  'vcloud' = Variance in clouds
  9.0    'vtair ' = Variance in air temperature
  2.e-17 'vprcp ' = Variance in precipitation
  0.0    'vrlhum' = Variance in relative humidity
  25.e4  'scorr ' = Horizontal radius of correlation (meters)
  2.0    'tcorr ' = Temporal radius of correlation (days)
  2      'prsflg' = 0 uncorr wind/slp, 1= wind from slp, 2=wind from slp limited by wndspd
```

In this file "randf" is used to switch on and off random forcing. Set it to 0 to switch off random forcing. The variance values denote the variance of different random forcing fields: Sea level pressure, air-drag coefficient in x, air-drag coefficient in y, maximum wind speed allowed of the perturbation, clouds, temperature of the air, precipitation, relative humidity. The correlation value "scorr" given in meters is used to set up the decorrelation length scale of the red noise. The decorrelation time scale of the red noise is set by the variable "tcorr" and is given in days. The final value indicates how to calculate wind and stress random forcing values. If "prsflg" is 0 the wind vectors are uncorrelated, while they are correlated using a geostrophic assumption when "prsflg" is set to 2. "prsflg" should not be set to 1, as it leads to too high wind/stress forcing.

5.1.5 infile.evp

This file is used to set up the EVP sea-ice dynamics model. It contains settings for EVP time step, and some parameters used to configure the sea-ice model, such as the ice strength and the ice concentration factor used in the original Hibler ice strength formulation. These parameters are also known as $C^\star$ and $P^\star$. The file looks like this:

```
1.1      # File version
 7200.   # EVP time step
27500.   # EVP ice strength
 20.     # EVP ice concentration factor
```

To use the EVP model, it must be enabled at compile-time with the EVP flag, see Section 4.2.4.

5.1.6 infile_gp.in

This is the configuration file for the generation of gridpoint data from HYCOM, which must be switched on in the file **infile.in**, see Section 5.1.3. This file makes it possible to select grid points based on longitude, latitude and depth ranges. It is also possible to subsample the grid, selecting only every n-th grid point along the grid dimensions, It is also possible to directly specify grid positions. An example of **infile_gp.in** is given here:

```
 1)     0.0          # minimum depth
 2)  9000.0          # maximum depth
 3)    50.0          # minimum lat
 4)    80.0          # maximum lat
 5)   -40.0          # minimum lon
 6)    40.0          # maximum lon
 7)       4          # skip in i index
 8)       4          # skip in j index
 9)  105   200       # Direct point specification
10)  200   200       # Direct point specification
...
```

In the example above, only points with depths between 0-9000 m, between 50° N and 80° N latitude, between 40° W and 40° E longitude will be included in the grid points processed. In addition, only every fourth grid

point (in each direction) will be processed. Finally, it is possible to specify an indefinite number of grid points directly by giving the i and j indexes (line 9 and 10 above). The grid point files are dumped in files with the position encoded into the file names. To avoid clutter, the files are placed in directory, one directory for a given month and year. The names of these directories are of the type **RRRGP_YYYY_MM/** where RRR is the rungen ID of the model, YYYY is the year and MM is the month. TODO: Netcdf output and processing.

5.1.7 Pre- and post-processing routines and job scripts

The pre and postprocessing scripts are named **preprocessing.sh** and **postprocessing.sh** - as mentioned before they are responsible for setting up model runs in the "scratch" directories and retrieving data from that directory and placing them in the "data" directory, see Section 5.1.2.

A sample job script **pbsjob.sh**, needed to run the model in a queue system, is given. This job script should work for a Cray XT4 system, using the PBS queue system. For other queue systems new job scripts need to be created, but the main steps for running the model are given by the following script segment(line numbers added):

```
 1 #Fetch OpenMP and MPI "sizes"
 2 NMPI=‘qstat -f $PBS_JOBID | awk ’/mppwidth/ {print $3}’‘
 3 NOMP=‘qstat -f $PBS_JOBID | awk ’/mppdepth/ {print $3}’‘
 4 [ -z "$NOMP" ] && NOMP=0
 5 export NOMP NMPI
 6
 7 # Enter directory from where the job was submitted - should be the experiment dir
 8 cd $PBS_O_WORKDIR       ||  { echo "Could not go to dir $PBS_O_WORKDIR  "; exit 1; }
 9
10 # Initialize environment (sets Scratch dir ($S), Data dir $D, experiment dir $P++ )
11 source ../REGION.src  || { echo "Could not source ../REGION.src "; exit 1; }
12 source ./EXPT.src  || { echo "Could not source EXPT.src"; exit 1; }
13
14 # Transfer data files to scratch - must be in "expt_XXX" dir for this script
15 ./preprocess.sh         ||  { echo "Preprocess had fatal errors "; exit 1; }
16
17 # Enter Scratch dir and  Run model
18 cd $S     ||  { echo "Could not go to dir $S "; exit 1; }
19 aprun -n $NMPI -m 1000M ./hycom  # Run hycom
20
21 # Cleanup and move data files to data directory - must be in "expt_XXX" dir for this script
22 cd $P     ||  { echo "Could not go to dir $P  "; exit 1; }
23 ./postprocess.sh
24 exit $?
```

The job must be submitted from the experiment directory in the above script. A description of the run process, in chronological order, is as follows:

1. In the segment above, lines 2-5 fetch the number of MPI threads to use, using information in the PBS queueing system on Cray XT4. The most important part is the retrieval of the number of MPI tasks. On another queue system this will have to be done differently. It is also possible to specify these numbers manually in the script, by substituting lines 2-3 with "NMPI=10" and "NOMP=1" (for instance).

2. Line 8 makes sure that we change directory to the experiment directory before proceeding. If these files can not be found an error is returned and the job script will exit[1].

3. Lines 11-12 source the region and experiment source files, in order to set up the environment for this experiment[1].

[1]Note that the commands after the || sign are done if the command before the it fails.

4. The preprocessing is done on line 15, which moves all the data needed to run the model into the "scratch" directory. If this fails the job script exits.

5. Lines 18-19 change directory to the scratch directory, and runs the model. Note that the "aprun" command is particular to the Cray XT4 system and will have to be changed for other systems (mpiexec and poe, for instance)[1].

6. Lines 18-19 change directory to the scratch directory, and runs the model.

7. When the model finishes we go back into the experiment directory and run the postprocessing script, on line 22 and 23[1].

8. On line 24 the script exits with the exit status of the last command (postprocess.sh).

5.2 Tidbits

This section contains some additional information for running the model

5.2.1 Activating outer nesting

5.2.2 Activating inner nesting

5.2.3 Initializing the model from climatology

The procedure for initializing a model from climatology is now more in line with standard HYCOM. In standard HYCOM the file **limits** is used to indicate integration limits, and if a model should be initialized from climatology. This file is not specified directly in NERSC-HYCOM, instead it is pre-processed using the contents of the file **infile.in**.

In order to run the model with climatology in NERSC-HYCOM, you should create an empty file called "INITIALIZE" in the experiment directory, which can be done like this

```
touch INITIALIZE
```

This file will be read by the preprocessing script, more specifically the script **subprogs/setlimits.py**. What it does is create the file **limits** in the scratch directory, with a negative start time, which indicates to HYCOM that the model should be initialized from climatology. The file "INITIALIZE" will be deleted after preprocessing, to avoid re-initializing the model at a later stage.

Note that in order to initialize the model, you should use climatological forcing in **infile.in** (set line 6 for example as: month era40), and you should set yearflag to 0 or 1 in **blkdat.input**.

Finally, you should not use a reference year of 0 in **infile.in** when initializing the model. I recommend using a reference year of 2 (line 3 is: 0002). This may seem strange but it is due to particularities of the reference dates used in standard HYCOM.

5.2.4 Starting the model from a restart file with "wrong date"

Sometimes it is necessary to initialize a model run from a restart file which has a different date. This is usually done by copying the restart file from, for instance, year 2002 to 1958 by modifying its filename. In this case the date information inside the restart file will not match the file name. This basically causes the model stop and complain that the date of the restart file is wrong.

In order to correct this, HYCOM needs to be told that it shouldnt consider the time inside the HYCOM restart, but that it should use the date specified in the file **infile.in**(and put into the **limits** file in the scratch directory) as the correct date. This is handled in a similar way as for the model initialization and therefore is only possible if yrflag=3 (using realistic synoptic forcing). By creating an empty file called "INITIALIZE" in the experiment directory, which can be done like this

```
touch INITIALIZE
```

HYCOM will ignore any date inconcistencies. Remember that this is done only when yrflg=3 in **blkdat.input**.

5.2.5 Starting the model from a curviint restart file

The curviint routine will interpolate fields from another model onto the current experiment model, as described in Section 3.8. If you want to start from a curviint-file created by the routines in Section 3.8, you can create an empty file called CURVIINT in the experiment directory, like this

touch CURVIINT

This will make the preprocessing script look for a curviint file in the region_basedir/curviint/$expt_number/ in the experiment.

5.2.6 Quick access to data and scratch directories

The experiment environment can be sourced to easily move to the scratch and data directories. By typing these lines (when standing in the experiment directory)

```
source ../REGION.src  || { echo "Could not source ../REGION.src "; exit 1; }
source ./EXPT.src  || { echo "Could not source EXPT.src"; exit 1; }
```

the scratch and data directories are placed in environment variables. To go to the scratch directory, type

cd $S

and to go to the data directory, type

cd $D

A similar approach is also used in the job scripts to set up the directories properly, see Section 5.1.7.

5.3 Running the model

Assuming all data files are set up properly, Section 3, The code is set up properly, Section 4, and the model run configuration files are set up properly, 5, the model should be ready for integration.

At this point you can check if all the files needed are present by running the preprocessing script **preprocess.sh**. This script can be outside of the queue system. If this script fails it means something is missing, and you need to modify your setup. The script should give you some indication on what went wrong, and some additional info is present in the **log/** directory.

When you are content with the setup, the model is ready to run. It can be submitted using the job script. On hexagon, this will run the model:

qsub pbsjob.sh

Chapter 6

Overview of auxillary routines and scripts

Several diagnostic and setup tools have been developed at NERSC, these are briefly described here. One of the main changes relative to previous versions of the NERSC diagnostic/setup tools is that they are now gathered in one location, with a common makefile setup. This makes porting the program code much easier.

Many of the routines used by several different diagnostic tools have also been put into libraries. Routines such as the conformal mapping tools, time processing tools, and routines for reading model grid and model restart, daily, weekly and archive files have all been moved into a library directory. This way duplication of work is avoided.

6.1 Main directory

The routines are gathered under the directory MSCProgs. On the svn, this directory is located at

./hycom/MSCProgs/

This means, that the program are most of the time independent of the HYCOM version. However, in some case the version matter, then there will be a different executable followed by the version number. The contents of that directory are

```
src
src_others
include
lib
Input
data
scripts
bin
bin_setup
```

The **src/** directory contains the code for the different routines which can be used to diagnose HYCOM output, plus various other convenient tools. The **src_others/** directory is meant to contain 3rd party tools, which are again used by the various tools located in the **src/** directory. Presently **src_others/** only contains the FES2004 tidal database routines. The **include/** and **lib/** directory will contain include and library files, respectively, which are used by the different routines in the **src/** directory.

The **Input/** directory is meant to contain examples of input files used by the different diagnostic and setup tools. The **data/** directory contains datasets. These are typically datasets of a relatively small size. The tools themselves are placed in the **bin/** and **bin_setup/** directories. The latter directory is, as the name implies, mainly meant for tools related to the setup of NERSC-HYCOM. The **scripts/** directory contains various useful scripts.

To compile the tools, any 3rd party tools in **src_others/** should be compiled first, and then the routines in the **src/** directory should be compiled.

Variable	Description
CF90	Denotes the FORTRAN90 compiler to use.
CF77	Denotes the FORTRAN compiler to use, usually the same as CF90.
CC	Denotes the C compiler to use.
LD	Denotes the linker to use, usually the same as CF90.
CPP	Denotes the c preprocessor to use.
F90FLG	Compiler flags for compiling free-source code (flag name is a bit misleading).
F77FLG	Compiler flags for compiling fixed-source code (flag name is a bit misleading).
FFLAGS	FORTRAN/FORTRAN90 compiler flags.
CFLAGS	C compiler flags.
CPPFLAGS	C preprocessor flags.
LINKFLAGS	Flags passed to the linker.
LIBS	Library flags (library search directory and actual library).
INCLUDE	Include flags (include search directory).
NCARGCF90	FORTRAN90 Compiler wrapper for NCAR Graphics.
NCARGCF77	FORTRAN Compiler wrapper for NCAR Graphics.
NCARGCC	C Compiler wrapper for NCAR Graphics.
NCARGLD	Linker wrapper for NCAR Graphics.
FFLAGSR4	Real*4 version of FORTRAN/FORTRAN90 flags.
CFLAGSR4	Real*4 version of C flags.

Table 6.1: Makefile macros set in **make.inc**.

6.2 The "src/" directory

6.2.1 Overview

The contents of the **src/** directory are designed to be relatively easy to port to different systems, with a minimal amount of hand configuration. There is a top-level makefile in this directory, which will perform the compilation in all the subdirectories under the **src/** directory. The makefile can also be used to install all the programs in the **src/** directory into the **bin/** and **bin_setup/** directories.

The configuration files for the makefile are located in the **Make.Inc/** subdirectory of **src/**. Proper setup of configuration files is needed to compile the different programs correctly.

6.2.2 Configuring the make include file

To prepare the include file needed to compile the different utilities, you need to set up a file called **make.inc** inside the **Make.Inc/** directory. This file will contain all the macro definitions needed for compiling the model. The macros are described in Table 6.1.

The CPP flags defined in the macro CPPFLAGS must be set properly for the machine and compilers you use. Table 6.2 gives the most common values for the CPP flags. Note that some of the flags do the same thing. The safe option is to define both the FFTW and the LAPACK flags and undefine the ESSL flag, which should work on most architectures. On some AIX (IBM) machines the Engineering Scientific Subroutine Library (ESSL) is also installed and can be used. If the ESSL flag is defined on these machines, you should undefine the FFTW and LAPACK CPP flags. Finally note that the IARGC flag may be needed for some compilers. Set it if you get messages about undefined iargc during compilation.

There are already several make include files inside the **Make.Inc/** directory, use these as starting points when setting up the model for a new machine. Once a suitable include file is created link or copy it to the file **make.inc**.

6.2.3 Compiling the code

After the make include file **make.inc** has been set up the code can be compiled. First make sure that you start with a clean slate by typing

CPP flag	Description
IARGC	Flag is used to denote if the FORTRAN iargc() function must be defined as external. This varies from compiler to compiler.
FFTW	Define this flag to use the "Fastest Fourier Transform in the West" routines. This popular GNU licensed FFT library is installed on most HPC machines. You must also define the libraries and include flags in the LIBS and INCLUDE macros.
LAPACK	Define this flag to use the LAPACK library. This is also installed "everywhere". You must also define the libraries and include flags in the LIBS and INCLUDE macros.
ESSL	Define this flag to use the "Engineering Scientific Subroutine Library" library, available on AIX(IBM) machines. If you define this, FFTW and LAPACK must be undefined in the CPP flags. You must also define the libraries and include flags in the LIBS and INCLUDE macros.

Table 6.2: CPP flags set in the CPPFLAGS macro in **make.inc**.

make clean

This is mainly needed when the code is ported to a new machine, or if the **make.inc** is changed. Normally it is not needed. To compile the code type

make all

and you should be greeted with compiler output flying over your terminal screen. If something fails, it can usually be corrected by making sure that the file **Make.Inc/make.inc** is properly set up. If compilation succeeds, the subdirectories of the **src/** directory should contain the compiled routines. To move them into the **bin/** and **bin_setup/** directories type

make install

If no errors occur during compilation, all the executable will be copied to the bin and bin_setup directory. Make sure that this is the case. Some folder that are rarely used or in development are removed in purpose from the Makefile. For example Note the NCAR Graphics wrappers and real*4 versions of the compiler flags are used to compile visualization routines inside the **NCARG-test/** directory. These can sometimes be hard to port to different machines. Don't be overly alarmed if these fail to compile on some machines, there are different diagnostics routines which do similar things. and the routines should be properly setup. They can now be easily accessed by setting the PATH variable of your shell ($MSCPROGS/bin and $MSCPROGS/bin_setup).

6.3 The "src/Nersclib/" directories and its libraries

The **Nersclib/** directory contain the libraries needed to compile most of the routines in the **src/** subdirectories. When you type "gmake all" in the **src/** directory, the contents of the **src/Nersclib/** will be compiled and the output is placed in the **include/** and **lib/** directories. When the other routines in the **src/** directories are compiled, they will use the contents of **include/** and **lib/** to gain access to the functionality in these libraries.

The routines inside the **Nersclib/** directory cover a range of common operations. Typically they are used for reading the model grid (longitude, latitude, depth and masks), working with the conformal mapping routines, and reading hycom files such as restart, daily average, weekly average and archive files. Some information is also available in the FORTRAN files under the **Nersclib/** directory. A rough overview of the functionality of the different routines in **Nersclib/** is given in Table 6.3.

The libraries installed in the **lib/** directories are

- The library **libconfmap.a**, which contains the functionality of the conformal mapping routines.

Fortran file in **Nersclib/**	Description
mod_confmap.F90	Contains functions and routines for using the conformal mapping. Includes grid transformations from lon,lat to i,j.
mod_grid.F90	Used to read model grid and topography.
mod_hycomfile_io.F90	Use to read fields from several hycom-type files
mod_parameters.F90	Contains a few commonly used parameters
mod_spline_calc.F90	Used to calculate vertical interpolation (spline, staircase and linear).
mod_xc.F	Low-level routine for initialising HYCOM grid size. Standard HYCOM routine.
mod_za.F	Low-level routine for reading HYCOM files. Standard HYCOM routine.
rotate.F90[1]	Simple routine for rotating grid between east/north directions and grid directions.
sort.F90[1]	Sorting routine.
spherdist.F90[1]	Routine for calculating sphere distance between two points.
machi_c.c[1,2]	Various machine-dependent routines.
machine.[1,2]	Various machine-dependent routines.
wtime.F[1,2]	Various machine-dependent routines.

[1] Does not need a use statement.
[2] Rarely needed, Mainly used by **mod_xc.F** and **mod_za.F**.

Table 6.3: Fortran files and the use statements and libraries needed to access functionality of the Fortran files.

- The library **libhycnersc.a**, which contains all other functionality in the fortran files under **Nersclib/**

Inside the **include/** directory are all the modules which are needed in conjunction with the different libraries. To use functions or subroutines associated with the different fortran files inside **Nersclib/**, Table 6.4 lists the combination of "use" statements and libraries needed for successfully compiling the code using these routines. Examples of the usage of the **Nersclib/** can be found in almost all of the routines located under the **src/** directory. Use these as staring points for their usage. Unfortunately the documentation is somewhat lacking apart from what is given in the FORTRAN files. A few pointers is given below for some of the most commonly used modules.

mod_za, mod_xc and mod_grid: Typically, the majority of the routines depend on functionality inside the modules mod_xc and mod_za. In almost all cases these should be initialized first, through these statements (in that order)

```
call xcspmd()
call zaiost()
```

which initialize the grid size in mod_xc and I/O setup in mod_za. The routine mod_grid is another frequently used module. After mod_xc and mod_za has been initialized, mod_grid will contain grid information (depth, longitude, latitude and more) after issuing the command

```
call get_grid()
```

Examples of its use can be found in most routines in the **src/** directory.

mod_spline_calc: The module mod_spline_calc is used for vertical interpolation. It needs an initialization statement before splines can be calculated. This initialization call sets the vertical levels to interpolate to. It is initialized either from a input file called **depthlevels.in** through the initialization call

```
call spline_calc_ini_fromfile(cmethod)
```

or it can be initialized from arguments

```
call spline_calc_ini_frominput(cmethod,deeps,ndeep)
```

Here, cmethod is a text string denoting the vertical interpolation method, whereas deeps and ndeeps is

Fortran file in **Nersclib/**	use statement needed	library needed
mod_confmap.F90	use mod_confmap	**libconfmap.a**
mod_grid.F90	use mod_grid	**libhycnersc.a**
mod_hycomfile_io.F90	use mod_hycomfile	**libhycnersc.a**
mod_parameters.F90	use mod_parameters	**libhycnersc.a**
mod_spline_calc.F90	use mod_spline_calc	**libhycnersc.a**
mod_xc.F	use mod_xc	**libhycnersc.a**
mod_za.F	use mod_za	**libhycnersc.a**
rotate.F90	–[1]	**libhycnersc.a**
sort.F90	–[1]	**libhycnersc.a**
spherdist.F90	–[1]	**libhycnersc.a**
machi_c.c	–[1,2]	**libhycnersc.a**
machine.F	–[1,2]	**libhycnersc.a**
wtime.F	–[1,2]	**libhycnersc.a**

[1] Does not need a use statement.
[2] Rarely needed, Mainly used by **mod_xc.F** and **mod_za.F**.

Table 6.4: Fortran files in **Nersclib/** and the use statements and libraries needed to access functionality of the Fortran files.

an array with ndeeps elements containing the vertical levels. TODO: clean up, refer to **depthlevels.in**. After that the various spline interpolation methods can be called. Examples of its usage is in the directory **src/Hyc2proj/**.

mod_hycomfile_io: The module mod_hycomfile_io is used for reading various hycom files(restart, averages, archives). It relies on the use of the hycomfile type. A typical way to initalize and read fields is

```
type(hycomfile) :: hfile
...
call initHF(hfile,filename,ftype)
call HFReadField(hfile,fld2D,idm,jdm,'saln',1,1)
```

which initializes the hfile type, based on filename and filetype (filetype can be either "restart", "nersc_weekly", "nersc_daily" or "archv"). It then reads salinity at depth level 1 and time level 1 into the "fld2D" array with dimensions (idm, jdm). An example of its use can be found in many places, for instance in **src/ExtractNC3D/**. There is also some additional info in the header of **Nersclib/mod_hycomfile_io.F90**.

mod_confmap: The module mod_confmap sets up the conformal mapping routines, and is used in several places throughout the **src/** directory. The conformal mapping routines also rely on an initialization statement

```
call initconfmap(idm,jdm)
```

which initializes the data inside the mod_confmap module. Here, idm and jdm are the horizontal grid size dimensions. The confmap routines are used to transform between horizontal grid indexes (i, j) and longitude, latitude values. An example of its use can be found inside the **src/ConfmapRoutines/** directory.

6.4 Quick explanation of the rest of "src/"

The following is a very short description of the directories under the **src/** directory. Most of these routines will require the grid/depth files of the models, many of them also need the conformal mapping file **grid.info**.

Note that additional info is available in most of these directories, through README files, but also in the routines themselves. Most routines which take arguments give additional info when run without arguments. In addition, some routines contain directories with matlab snippets for reading/diagnosing results of the routines.

6.4.1 Average - Create 3D averages of HYCOM files

This directory is for creating the routine **hycave**, which can be used to calculate "averages" of hycom files. The final file will have XXX for rungen and 9999 for year 99 for month or 999 for day. (example: XXXAVE9999_999_9.a or XXXDAILY9999_999_99.a) Run with no arguments for a short description of its usage.

6.4.2 Barstrf - Create Barotropic streamfunction

This directory is for the routine **hyc2bastrf**, which is used to calculate the barotropic stream function. Run with no arguments for a short description of its usage.

6.4.3 Conf_grid - generation of model grids

This routine is used to generate model grids using the conformal mapping tools. It creates the regional grid/depth files which are used by HYCOM during model runs. Some guidelines for generating model grids with this tool is given in Section B.

6.4.4 ConfmapRoutines - Going between grid indexes and geographical positions

This directory contains routines for going from longitude latitude to grid indexes, using the conformal mapping tools.

6.4.5 Copymem - copies ensemble members

Creates routine **copymem**, used to copy from one ensemble member to another. Mainly used during data assimilation.

6.4.6 Curviint - restart file interpolation

Creates the executable **curviint**. Used to interpolate restart files from a model onto another model grid. Help is available. This routine is also used by the setup routine **curviint.sh**, see Section 3.8.

6.4.7 DateTools - convert between ordinal day to real date

Routines used to go between real dates and ordinal days relative to a reference date. Uses the Gregorian calendar only.

6.4.8 DProfile - Vertical profiles.

Creates the routine **dprofile**, which is used to extract vertical profiles from model files. Run with no arguments for more info. This routine uses the "extract" files to specify fields to extract from the various HYCOM files, see Section 6.5.1 for more info on this file format.

6.4.9 Ensstat - calculates ensemble/time statistics

Various routines used to calculate ensemble statistics. Can also be used with daily and weekly files. Routines can be run without arguments for usage information. Some routines need the "nco" tools to work. Note that this routine can do ensemble statistics as well as time-domain statistics.

6.4.10 ExtractNC2D - Convert from hycom files to 2D netcdf files

Creates routines for conversion between hycom files and netcdf files with 2D fields. Main routine is the script **m2nc**, which is used to process input arguments etc. Run **m2nc** with no arguments for more info. The routines needs input files of the "extract" type, described in Section 6.5.1.

6.4.11 ExtractNC3D - Convert from hycom files to 3D netcdf fields

Creates routine **h2nc** for conversion between hycom files and netcdf files with 3D fields. Run **h2nc** with no arguments for more info. Needs input files of the "extract" type, described in Section 6.5.1.

6.4.12 FindLayer - find layer thickness between threshoolds

Routine **findlayer** calculates layer thickness between two levels of a variable, for instance layer thickness of salinity between 35 and 36 psu. Run with no arguments for more information.

6.4.13 GP - extract data from grid point files

Various routines for extracting data from GP time series files. Routines can be run with no arguments for more info. Activating GP timeseries in NERSC-HYCOM is done in **infile.in**, See Section 5.1.3.

6.4.14 GPdens - grid point time series statistics

Routine **gpdens** calculates statistics of a grid point time series at a specified depth level. The focus is on current statistics, calculating speed density, exceedence plots, tidal decomposition etc. The routine **gpdens** can be run with no arguments for more info. Activating GP timeseries in NERSC-HYCOM is done in **infile.in**, See Section 5.1.3.

6.4.15 GridToLL - interpolation to regular lon/lat grid

Test routine **gridtoll** for interpolating fields from NetCDF files (tmp1.nc, from Section Section 6.4.10) to a regular grid. example: gridtoll -98:.1:20 10:.1:60 temp01
For extracting the temperature of the first layer into a grid with 1:20 and 1:60 discretization.

6.4.16 Hyc2proj - interpolation to projection grid and z-levels

This directory contains the routines **hyc2proj** and **hyc2stations**, two routines which can be used to interpolate HYCOM model results. The routine **hyc2proj** interpolates HYCOM fields to a map projection and fixed z-levels, while **hyc2stations** interpolates HYCOM fields to spatial locations and fixed z-levels. The routine **hyc2stations** can be used to calculate vertical cross sections, for instance. Both routines produce NetCDF files as final output. The routine **hyc2proj** is a interpolation routine which takes command-line arguments, run it with no arguments for a brief summary of the options. In addition it requires a few input files to run. These files are

- **proj.in** to specify the horizontal projection to use, see Section6.5.2.
- **depthlevels.in** to specify the depth levels to interpolate to, see Section6.5.4
- "extract" files to specify fields to extract from restart, daily and weekly average , see Section6.5.1.

These input files will be read by the **hyc2proj** routine, and the projection to use (from proj.in), the depth levels to interpolate to (from depthlevel.in), and the variables to interpolate will be set accordingly (extract.???). The interpolation itself is a two-stage process. It will first interpolate the model fields horizontally, using a bi-linear interpolation method. When that is done the ocean 3D fields are still on the hybrid vertical grid. The vertical interpolation is then done, using a vertical cubic spline method. There are also additional vertical interpolation methods that can be used, such as "staircase" - which uses constant values when the z-levels to interpolate to are positioned between the lower and upper interfaces of a model layer. There is also a "linear" option which uses linear interpolation between the midpoint of model layers. Both the "staircase" and "linear" methods are faster than the default spline method, but the spline method often gives better/nicer looking results.

The routine **hyc2stations** is used to interpolate model fields onto specified spatial positions. The positions are grouped together, which can be used to define "sections". It also takes command-line arguments, run with no arguments for a brief description of these. To use the routine the following input files are needed

- **stations.in** to specify the points to interpolate to and the group definitions, see Section6.5.3.
- **depthlevels.in** to specify the depth levels to interpolate to, see Section6.5.4
- "extract" files to specify fields to extract, see Section6.5.1.

These input files are read by **hyc2stations** to set up the stations to interpolate to, the vertical depth levels and fields to extract. As with **hyc2proj** the interpolation is done in two stages. First bi-linear interpolation to the station positions, and then cubic splines in the vertical. The routine does not support linear or staircase vertical interpolation, as **hyc2proj** does.

The **hyc2stations** routine will organize the stations into "groups", which is a way to organize the stations into disjoint sets. This way one can group stations together to form vertical cross-sections, for instance. This grouping is defined in the input file **stations.in**, described in Section 6.5.3. Normally the hyc2stations routine produces one netcdf file per group defined. However, it is possibe to modify this behaviour by setting the CPP FLAG "MULT_GRP_PER_FILE" in the makefile in the **Hyc2proj/** directory. This way only one file will be created, containing all the groups.

Also note that there is a small python script **setupstations.py** in this directory, which can be used as an aid when setting up stations used in **hyc2stations**. By specifying start and end points this routine will create stations between these two points, in a format suitable for inclusion in **stations.in**. These intermediate points are created following rhumb lines (not great circles).

6.4.17 IceDrift - calculation of sea-ice drift

This directory contains a routine for calculating ice drift. Input is DAILY average files, or specially formatted ensemble ice drift files. Not very well documented. The CPP FLAG EVP_DRIFT_ASSIMILATION must be set in NERSC-HYCOM to produce ensemble drift files.

6.4.18 Idealized_Grid - Setting up idealized grids and restart files

The routines in this directory can be used to set up idealized grids and restart files, which can be useful to test model behaviour. These files are only meant as starting points, you will need some hand-editing to set up the configuration you need. Basic cases such as seamount and flat-bottom channel are at least partially implemented.

6.4.19 InterpTest - unfinished interpolation "toolchain"

Preliminary work on an interpolation "toolchain", where routines pipe result from one program to another. Needs more work.

6.4.20 MkEnsemble - creating a model ensemble

The executable is called **mkensemble**. This routine can create a model ensemble. This is done by perturbing the layer thickness in the hycom restart files, as well as perturbing the ice thickness fields in any ice restart files. Input are given in the file **mkensemble.in**. The routine are working diffrently in 2.2.12 and 2.2.37. To get more info on how to run it with 2.2.12 run the executable without **mkensemble.in**. In version 2.2.37, this file will typically look like that:

```
TP4restart2011_249_00_mem
TP4restart1990_247_00ICE.uf
1               # Ensemble mem we are reading from
2               # Ensemble mem we are dumping in
0.05          # Std dev of log(d), no unit, 0.1 = 10\%
3             # Vertical correlation range   (layers)
10            # Horizontal correlation range (grid cells)
```

In the first line, you specify the basename of the input file. In the second line you specify the ice input file (in the example they are not from the same initial file although this is recommended). Then you need to give the member you are reading from and to which member you are dumping the member into. It is possible to have the same input and output member. Then follows specification about amplitude of the perturbation and the spatial correlation. In the example her, mkensemble will read the physic from TP4restart2011_249_00_mem001.[a,b] and dump it into PERTTP4restart2011_249_00_mem002.[a,b]. If ice record are present in the a,b files (i.e. if the FLAG SINGLE_RESTART is activated) perturbed ice thickness will also be dumped. Beware that when HYCOM is running if SINGLE_RESTART_ONLY FLAG is not

activated, the ice record in the a,b files have lower priority than the one in the ICE.uf. In addition, the program read the record corresponding to the first member in TP4restart1990_247_00ICE.uf and dump the ice fields with perturbed ice thickness into PERTURBED_ICE.uf. If you don't have ice files or you don't want to perturb the ice, set "none" in the second line.

6.4.21 Model_input-x.x.xx: utility file

This directory contains some program to create input files necessary for HYCOM when some specification are used. As these file are formatted differently depending of the version of HYCOM there is one folder for each version. In 2.2.12 the only program is the one that produce the file for spatially varying thickness diffusion. Indeed, when negative value is given for thkdf4 in the blkdat.input, hycom will read a spatially varying value from a file call thkdf4.[a,b]. The reason for using spatially varying value is that thkdf4 can degrade quite strongly the stratification and should be only used when it is necessary (noise in the layer thickness below the mixed layer depth or crashes due to neg. dp). In TOPAZ we notice that thkdf4 was needed in the Gulf Stream area, but had a very bad impact at high latitude. Lower values are used there.

In 2.2.37, the program for creating spatially varying thickness diffusion is tuned for TOPAZ with the quick scheme. We use lower values in the Arctic, but needed to increase the values near the equator. There is also a different file for creating a file called sssrmx.[a,b]. This is needed when you don't want to use relaxation of salinity when you are too far away from climatology. Currently we use a constant calue of 0.5 meaning that it is strictly equivalent to defining the MATS_RELAX flags in 2.2.12.

6.4.22 Multiproc_New - Calculate hycom average and mean SSH fields

This directory compiles the routine **multiproc**, another routine that can be used to calculate average fields from hycom. The output fields are dumped in the old "pakk" file format. Avoid using this routine, and use the more generic routine **hycave** in stead, see Section 6.4.1.

The routine calculates the mean ssh fields, which are used for data assimilation purposes.

6.4.23 NCARG-test - visualization routines

If you can get NCAR graphics to install and compile, this directory contains routines which can be used to visualize the hycom model fields. Somewhat unfinished, but can be used to create quite nice plots.

Input files **cplot.in** and **cvplot.in** are needed for running the routines **cplot** (contour plot) and **cvplot** (contour + vector plot) in this directory. Note that there is no vertical interpolation done in the routines.

Files **cplot.in** and **cvplot.in** are briefly described in Section 6.5.8.

6.4.24 Nestbat - smoothing bathymetry towards an "outer" model

This directory contains two versions of the nestbat routines. One for version 2.1 of NERSC-HYCOM, and another for version 2.2. The version 2.2 file **nestbat-2.2** is used by the setup script **nestbat.sh** described in Section 3.2.

Information is available for both of these routines when starting the programs.

6.4.25 Nesting-2.2

These are nesting routines needed to set up version 2.2 of NERSC-HYCOM. The routines are **nestini** which prepares the nesting relaxation masks, **nestports**, which sets up the ports file needed for barotropic nesting in NERSC-HYCOM 2.2. The routine **nestpivots** is used to set up the outer model for nesting, by setting up details of the interpolation from the outer model grid to the inner grid.

6.4.26 Nest_Offline-2.1

This directory contains nesting routines which were used in version 2.1 of NERSC-HYCOM in order to produce file from an outer model. This is a work in progress , but interpolation from mercator or HYCOM with different stratification should be possible. TODO:Should be ported to version 2.2.....

6.4.27 NORSEXClim - climatology from NORSEX ice concentration

Program can be used to calculate a NORSEX ice concentration climatology, and interpolate it to a projection grid, similar that of **hyc2proj**, see Section 6.4.16. Note that some of the projections defined in **hyc2proj** are not available for this routine (usesold version of mod_toproj.F90 in **Hyc2proj/**). Also, the NORSEX files read are ascii files produced at NIERSC (I think).

TODO: In short, this routine may require some updates. Routine **mod_toproj.F90** may be updated and moved to **Nersclib/** since various tools use it.

6.4.28 ObsCompare - compare model with observations

This routine contains several routines for comparing observations and models. The main routines are

- Routine **argocmp**, which reads ARGO NetCDF files containing temperature and salinity profiles. These are compared wih HYCOM files, and the result of the comparison is dumped to text files. It is also possible to parse the HYCOM data into the ARGO NetCDF file.
- Routine **argo_extract_profiles_new.sh**, which is a shell script. Input is a HYCOM file (usually daily average. It looks for ARGO files matching that date, downloads them if necessary and runs the **argocmp** program.
- Routine **argoreduce** is a routine which can be used to calculate regional statistics from the ARGO vs model comparisons. Operates on text output from the **argocmp** routine. Needs the input file **regiondefs.in**, described in Section 6.5.7.
- Routine **slacmp** can be used to compare SLA from SSALTO/DUACS with the SLA from the model. In theory this routine can be compiled with OpenDAP support, but presently it needs local netcdf files for the comparison. Also needs the file **regiondefs.in** if regional statistics are to be calculated, The format of this file is described in Section 6.5.7.
- Routine **slacmp_ftp.sh** is a shell script which takes a hycom (daily) file as input. From this file a date is calculated, and the corresponding SLA file from SSALTO/DUACS is downloaded. The routine **slacmp** described above is then run.

Most of these routines have extra information in them when called with no arguments, in addition there is a README file in the source directory with some information.

6.4.29 Old_Forcing - Routines for creating "old" forcing files.

This directory contains the source code for generating forcing based on various data sets. The main routine is called **old_newf** (...) , and the routine is used by some of the forcing scripts when initializing NERSC-HYCOM forcing, see Section 3.5.

The data set used by this routine is relatively small and located in the **data/** subdirectory of the "MSCProgs" main directory. The "Old" in the name should tell you that this data is from an old data set. Avoid using it.

6.4.30 Old_Levitus - Routines for creating "old" SST and SSS fields

This directory contains the source code for generating forcing based on various data sets. The main routine is called **old_levini**, and the routine is used by some of the forcing scripts when initializing NERSC-HYCOM forcing, see Section 3.5.

The data set used by this routine is relatively small and located in the **data/** subdirectory of the "MSCProgs" main directory. The "Old" in the name should tell you that this data is from an old data set, with (among other things) large errors in the Arctic. Avoid using it.

6.4.31 Relax - model relaxation utilities

Contains the routine **rmunew** which can be used to create a relaxation mask in HYCOM. The routine is interactive, requiring input as it runs. This routine is used by the **relax_rmu.sh** script during the relaxation setup of HYCOM, see Section 3.4. TODO: make this use depthlevel.in as input.

6.4.32 RelaxToNetCDF - Get relaxation fields into NetCDF files

The routine **PHCtomodel.sh** can be used to interpolate from z-level climatologies used when initializing HYCOM, onto projection and fixed depth levels. The depth are currently hardcoded in the script. The aproach used is similar to that of the **hyc2proj** routine, see Section 6.4.16. It needs the input file **proj.in** (see section 6.5.2).

6.4.33 River_Forcing - prepare river forcing from point sources

The routine **rivers** is used to convert from point source river data files to a two-dimensional field, spreading the discharge along the model coast. Input files give point river data, specified in a file called **rivers.dat**.

The **rivers** routine can be called with no arguments to get some more information.

6.4.34 Section - Extract section data and transports

The **Section/** subdirectory contain routines for extracting data from sections to NetCDF files, and routines for calculating transport estimates from hycom files. There are several programs in this directory, but their functionality is most easily accessed using three different scripts; they are **m2section**, **m2transport** and **m2transport2**.

All the scripts in this directory will first call a routine **section_intersect**, whose purpose is to set up the grid points which are needed to calculate transports and to interpolate section details. The sections are specified in the file **sections.in**, see Section 6.5.5. Once a section is read, the grid points along a section are calculated.

The first step in this calculation is to locate the points which are on one side of the section. The sections follow great circles between the two points defining the start and end points of the section. Let these two points be called $\vec{r}_1$ and $\vec{r}_2$ in cartesian coordinates, and let the grid points in cartesian coordinages be given as $\vec{r}_{ij}$. Also, remember that the line segment from $\vec{r}_1$ to $\vec{r}_2$ is a part of a great circle, so it "splits" the earth into two hemispheres. All grid points $\vec{r}_{i,j}$ on one side of the hemisphere defined by the great circle must satisfy this criterion:

$$A = \{\vec{r}_{i,j} \text{ such that } (\vec{r}_1 \times \vec{r}_2) \cdot \vec{r}_{i,j} > 0\} \ . \tag{6.1}$$

The resulting set A of model grid points is illustrated in Figure 6.1b by the grey squares. Secondly, all points on the edges of the domain must be retrieved, a step which is done seperately for the u and v positions of the C-grid. Let $1_A(\vec{r}_{i,j})$ be the indicator function of set A, defined as

$$1_A(\vec{r}_{i,j}) = \begin{cases} 1 & \text{if } \vec{r}_{i,j} \in A \\ 0 & \text{if } \vec{r}_{i,j} \notin A \end{cases} \ . \tag{6.2}$$

All points on the edges of the domain defined by the set A can then be retrieved by the following procedure

$$B_u = \{\vec{r}_{i,j} \in A \text{ such that } f(\vec{r}_{i,j}) = 1_A(\vec{r}_{i,j}) - 1_A(\vec{r}_{i+1,j}) \neq 0\} \ , \tag{6.3}$$

$$B_v = \{\vec{r}_{i,j} \in A \text{ such that } g(\vec{r}_{i,j}) = 1_A(\vec{r}_{i,j}) - 1_A(\vec{r}_{i,j+1}) \neq 0\} \ . \tag{6.4}$$

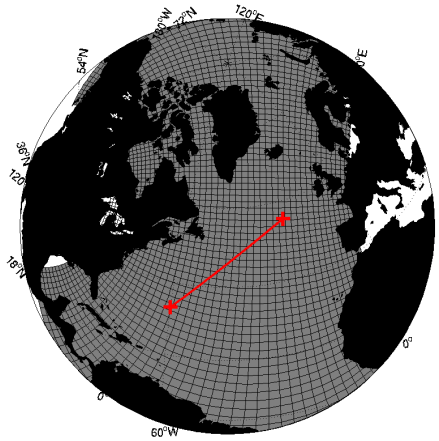
The resulting set B_u is illustrated in Figure 6.1c. Note that in addition to being used as a mask for points close to the boundary, the functions $f(\vec{r}_{i,j})$ and $g(\vec{r}_{i,j})$ will also have information on directionality. This is used to set up positive transport directions in the routines **m2transport** and **m2transport2**. Note that the set B_u and B_c will contain points which are on the great circle, but outside of the segment $r1$, $r2$. We can remove these by these conditions

$$C_u = \{\vec{r}_{i,j} \in B_u \text{ such that } (\vec{r}_1 \times \vec{r}_{i,j}) \cdot (\vec{r}_{i,j} \times \vec{r}_2) > 0\} \ , \tag{6.5}$$

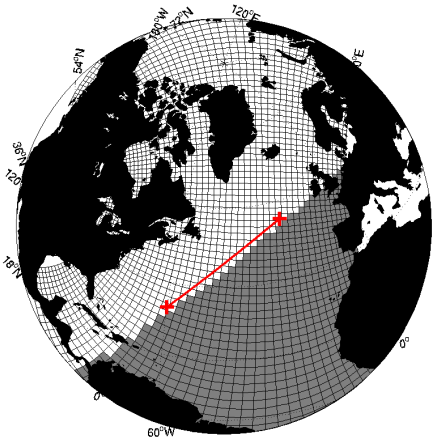
$$C_v = \{\vec{r}_{i,j} \in C_u \text{ such that } (\vec{r}_1 \times \vec{r}_{i,j}) \cdot (\vec{r}_{i,j} \times \vec{r}_2) > 0\} \ . \tag{6.6}$$

The final sets of grid points C_u and C_v can now be used for plotting purposes, and for calculating transports in conjunction with the indicator functions $f_{i,j}$ and $g_{i,j}$. An illustration of the set C_u is given in Figure 6.1d.

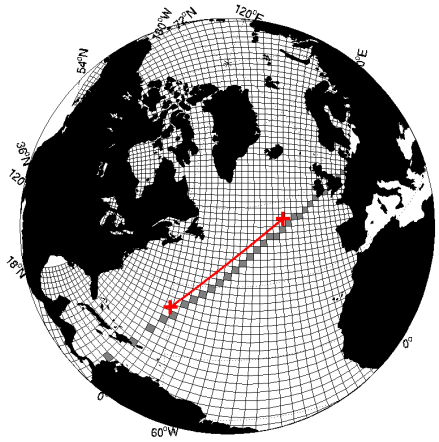
The functionality of the FORTRAN routines can most easily be accessed by running the various scripts inside the **Sections/** directory. These scripts are also installed inside the **bin/** directory. The routines are



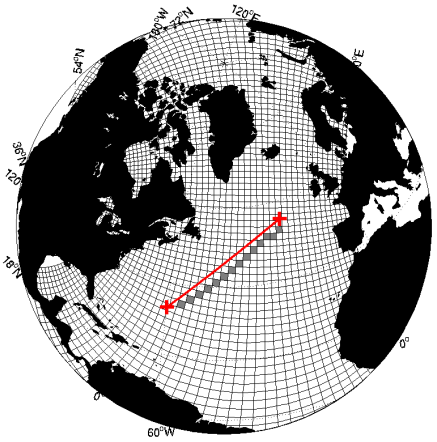
(a) Initial section with great circle between points $\vec{r}_1$ $\vec{r}_2$



(b) Points on one side of great circle excluded from set



(c) Remaining set after application of indicator function



(d) Final set after excluding points outside of line segment $r\vec{r}_1, \vec{r}_2$

Figure 6.1: The stepwise process of extracting section points, used by **m2section**, **m2transport**, and **m2transport2**.

- **m2section** - this script can be used to extract data along a section specified in the file **sections.in**, see Section 6.5.5. To specify which fields to extract, use the "extract" files described in section 6.5.1. This routine is run with the files to process as input arguments. Run **m2nc** with no arguments for additional information. Sections are dumped to files **section001.nc**, **section002.nc** ... There are matlab routines in the subfolder that can easily show a map of the section on a map (showsection.m) and plot easily the section with several option (sectionplot.m).

- **m2transport** - this script will calculate barotropic transports across a section. It will also try to calculate ice transports. Output is in ascii files named **transport_net001.dat transport_net002.dat**. As it only need to read the 2D field barotropic velocity, it is really fast. There is also an option to dump to a netcdf file called **transports.nc** when you compile the code. You will need to specify the sections in the file **sections.in**, see Section 6.5.5. Run **m2transport** with no arguments for short info on its usage.

- **m2transport2** - this script will calculate transports across a section, but with a more fine-grained approach than **m2transport**. You will need to specify the sections in the file **sections.in**, see Section 6.5.5. You will also have to define how to calculate transports in **transports.in**, See Section 6.5.6, where you can calculate transport between two depth levels, transport of water masses between two temperature levels , and more. It is also possible to use several critera at the same time. Run

m2transport2 with no arguments for short info on its usage. Files are dumped to ascii files with names defined in **transport.in**, and also the file **transports2.nc**.

- **m2transport2** with scalar transport. With **m2transport2** you can also calculate scalar transports in addition to volume transports. The routine **m2transport2** will look for a file **scalartransports.in** which can be used to calculate these transports, such as heat transport. The format of **scalartransports.in** is described in Section 6.5.6

The section code is re-written and should be more robust than in earlier versions. The calculation of points along the sections is also done in a more straightforward way.

When calculating the transports and sections, the files **section001.dat**, **section002.dat** ... will contain information on the points along the sections. The files **section001.dat**, **section002.dat** ... will contain information on the transport points and positive directions defined for the sections. The positive directions of a section can be found by the following rule. When standing at the first point of a section, and looking towards the next point of the section, then the positive direction is towards your right-hand side.

Remember that the sections follow great circle/geodesic lines on the sphere, not "straight lines" in longitude, latitude space (aka rhumb lines).

6.4.35 SSHFromState - calculate SSH from restart files

The small routine **ssh_from_restart** calculates the ssh fields from data in the restart files. Mainly used for assimilation purposes. Run with no arguments for short info on its usage.

6.4.36 Synoptic_Forcing-X.X.X- Creating forcing fields for HYCOM

The routines in this directory are used to generate forcing fields for HYCOM. The function involves the horizontal interpolation of forcing fields from several different data sets, the application of parameterizations to create new atmospheric forcing variables from the raw data sets, and finally saving the resulting atmospheric forcing fields to a file format which can be read by HYCOM. The raw datasets which can be used to produce HYCOM forcing are various ECMWF operational data sets, the ERA40 reanalysis, ERA interim reanalysis, the NCEP reanalysis and some other data sets. There are 3 different folders for this routine. The version 2.1 is kept but is deprecated. The difference between version 2.2.12 and 2.2.37 is that in the latter, the code is developed so that there are better agreement with the forcing produced with inline forcing. This means that a single routine is used for making the interpolation. It is worth noticing that when the forcing are preprocess, it is not possible to use KARA2002 implementation because the latter set the coefficient drag dependent on the discrepancies between the atmospheric surface temperature and the sea surface temperature (which is not yet available). However, a formulation called KARALIGHT has been set up so that you can still benefit from more accurate approach that computes the coefficient drag from the wind speed.

The directory consists of these routines

- **forfun_nersc-2.2.12** or **forfun_nersc-2.2.37** calculates the forcing fields needed to run NERSC-HYCOM. It is typically called like this

  forfun_nersc-2.2 synoptic_option climatology_option

  where the synoptic forcing option and climate forcing option can be any of the options given in Table 5.1, with the exception of the "prep" option for synoptic forcing. If the routine is run with synoptic forcing fields (i.e the synoptic option is anything apart from "month"), then the file **infile.in** will be parsed to find the start and end times of the integration, and synoptic forcing for that range will be produced. If called with synoptic option as "month" monthly climatologies will be calculated. In that case **infile.in** is not needed.

- **force_perturb-2.2** can be used to add random forcing fields to an already existing forcing data set. Not really necessary at the moment. Still as work in progress (not compile by the main makefile)

- **ncep_clim** is used to calculate a climatology from the NCEP data set. This climatology can then be used by the **forfun_nersc-2.2** routine. There is also a routine **get_ncep_ftp.sh** for downloading these data - which are continually updated.

In addition there is also a README file with some additional info. Note that the routine **forfun_nersc-2.2** is used in the scripts which set up the NERSC-HYCOM datasets (**nersc_clim.sh** and **nersc_synoptic.sh**) in Section 3.5.

6.4.37 Tides_CSR - CSR tidal forcing

This directory contains the routine **csr2mod_GE** which calculates tidal boundary forcing used in NERSC-HYCOM. This routine has been modified from previous versions, and uses the regional grid and depth files to set the grid size and calculate the forcing. Diagnostic output is also dumped in a NetCDF file. This routine is used to set up the tidal forcing in **nersc_csr.sh**, see Section 3.7.

6.4.38 Tides_FES - FES tidal forcing

This directory contains the routine **csr2mod_GE** which calculates tidal boundary forcing used in NERSC-HYCOM. This routine has been modified from previous versions, and uses the regional grid and depth files to set the grid size and calculate the forcing. Diagnostic output is also dumped in a NetCDF file. This routine is used to set up the tidal forcing in **nersc_fes.sh**, see Section 3.7.

6.4.39 TRIP - river forcing

The routines in this directory can be used to set up river forcing by using routines from the TRIP database [Oki and Sund], 1986, combined with runoff data from ERA40 or ERA-i. The TRIP database is used to calculate the paths of surface water flow, while the runoff data gives new water added to the surface. You need to define which TRIP data set to used in the makefile (TRIP05 is better). There are four routines in this directory.

- **trip_path** - a diagnostic routine which calculates paths and prints some information in text files. Identifies among other things the major runoff catchment basins and their associated river outlets.
- **trip_weights** - routine calculates the mapping from the ERA40 runoff grid to the TRIP dat. Needed to position the runoff data on the TRIP grid.
- **trip_flow** - routine calculates the flow of water from their origin toward the river outlets. The flow directions are given on the TRIP grid. Output is dumped to netcdf files. The routine **trip_weights** must be run before running this routine.
- **trip_tohycom** - the only really HYCOM-specific routine. Uses output from **trip_flow** to calculate river climatologies for HYCOM. With some minor changes to this routine and to NERSC-HYCOM itself, it should be possible to modify this routine to create synoptic fields of river discharge. Here, you may need to adjust the search radius or the land radius depending on the model resolution. There is a search radius that can be large (typically 300 km) and you need to define also the ellipse on which you are reducing the salinity from the river delta. (along shore radius and across shore radius).

6.4.40 ZONAL

This directory contains routines for creating zonal averages (**zonal**) and meridional overturning stream function(**mosf**). The latter needs some work to function properly.

6.5 The "Input/" directory - examples of input files

This directory contains some examples of input files needed to run the diagnostic tools.

6.5.1 "extract" files

Several routines use the extract files to read which files are to be processed. These files are different for different file types, and they are also processed differently by various routines. A typical extract file for restart files **extract.restart** looks somewhat like this (line numbers added)

```
 1: 4                      # Version of tecconv program
 2: dp       dp            # Field used to determine layer interfaces
 3: 2 T T T                # 2-D, sphere, rotate, index velocities
 4: 1 270 1 210            # section plot ia:ib ja:jb
 5: 1 1   XX XX   X        # extracted layers for 3-d fields (not used for 2-d)
 6: saln    01 01   T
 7: temp    01 10   T
 8: ficem   00 00   T
 9: hicem   00 00   T
10: ssh     00 00   T
15: ubavg   00 00   T
16: vbavg   00 00   T
17: pbavg   00 00   T
21: dpmixl  00 00   T
```

The first line in this file is a version number that can be changed when the format changes. The second line used to denote the field which determines the layer interfaces, but no routines in in the MSCProgs subdirectories use this anymore.

The third line is mainly used by the routines **m2nc** and **h2nc**. The first number is not really used anymore, whereas the 2nd to fourth logical values denote whether to dump velocities on original model grid, whether they should also be dumped to east/west components, and finally if they should be dumped th sphere velocities(3D vectors).

Lines 4 and 5 can also be ignored. From line 6 and onward are the specifications of fields to extract. The format is first field name to extract, the range of layers to extract for and finally a logical value which denotes if the fields should be extracted at all. The different routines do different things with these processing lines:

- The routines **m2nc** and **h2nc** will use all the information when exctracting data to netcdf files.
- The section routine **m2section** and the depth profile routine **dprofile** will just use the field name and the processing flag. All alyers are extracted.
- The interpolation routine **hyc2proj** will just use the field name and processing flag as well, because the model will be interpolated to fixed z-levels.

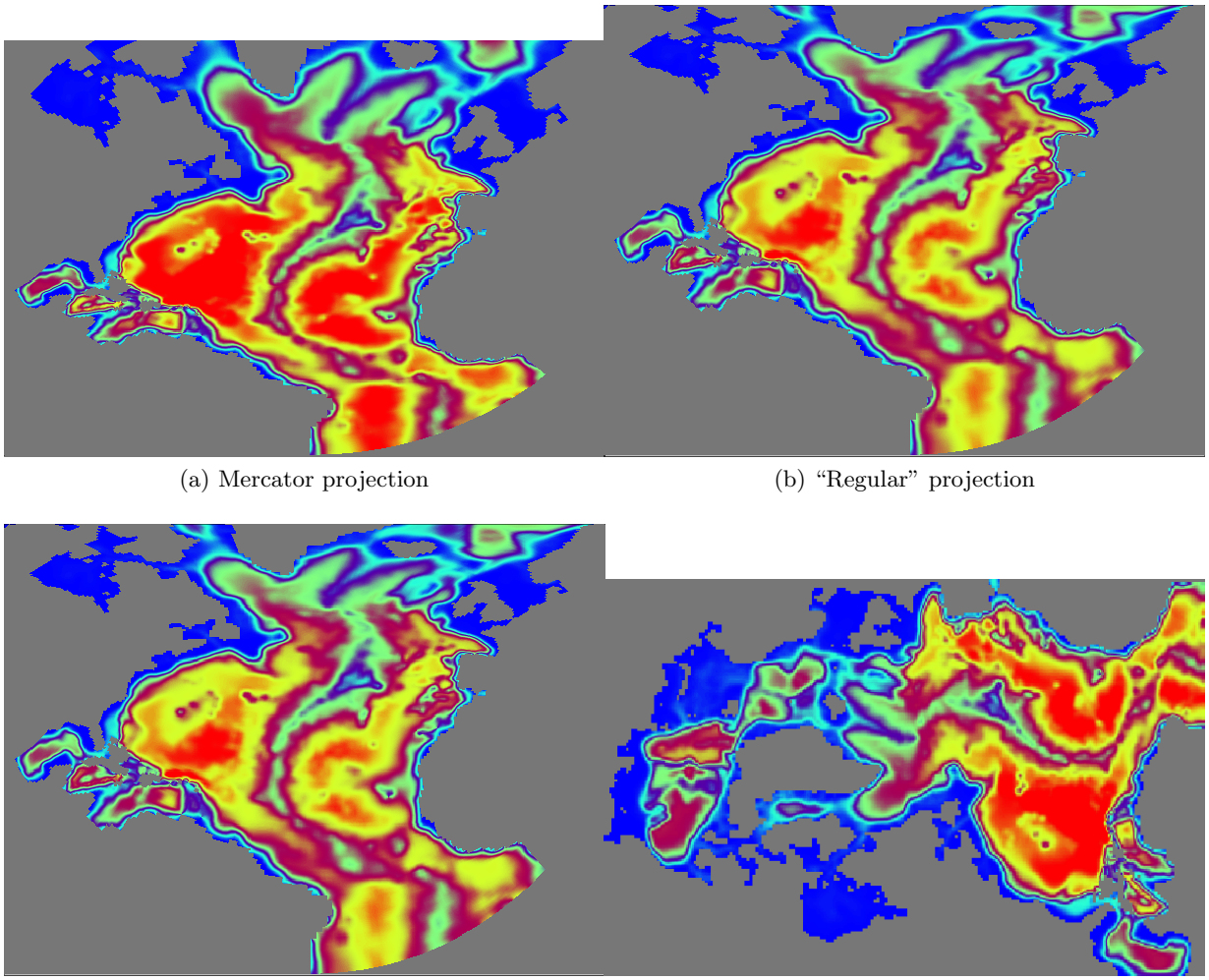
The different HYCOM files contain different fields, and names of fields which are the same (for instance salinity) can vary across different files. Due to this, there are different extract files for different file types. They are named **extract.restart**, **extract.archv**, **extract.daily**, and **extract.weekly**. Examples of these files can be found in the **Input/** directory. The field names in these files correspond to field name entries in the respective ".b" files of the .a/.b file pairs. If ou are missing a field name which you want to extract you can look in the ".b" file for the field name - if it is present you can add a new line with this field name to the extract files.

6.5.2 Projection file "proj.in"

The projection file **proj.in** can be used to interpolate from the model grid to a given projection. Available projections are "polar_stereographic", "regular", "mercator" and "native". The regular projection is not really a projection, it is a uniform, rectilinear longitude latitude grid. Similarly with the "native" option which is just the native model grid. Note that the polar stereographic projection in reality is a a stereographic projection (ahemm..), and can be defined anywhere on the sphere. An illustration of the different projections can be seen in Figure 6.2

There are example **proj.in** files for the different projections in th **Input/** directory. The input file **proj.in** for the various projections is described below.

"regular" projection option: This option will create a projection grid which is rectilinear in longitude-latitude space. With this option, the input file **proj.in** should look somewhat like this (line numbers added):



(a) Mercator projection

(b) "Regular" projection

(c) Stereographic projection

(d) "native" projection

Figure 6.2: Different projections which can be specified by **proj.in**.

```
1: regular
2: -30.0              # Longitude of W edge
3:  82.0              # Longitude of E edge
4: 0.200              # Longitude grid spacing
5:  45.0              # Latitude of S edge
6:  88.0              # Latitude of N edge
7: 0.100              # Latitude grid spacing
```

The first line denotes that the "regular" projection method is used. Lines 2-4 denote the first longitude point to interpolate to, the last longitude point to interpolate to, and the spacing between longitude points. The latitude values are given in a similar manner on lines 5-7.

The number of longitude and latitude points in the example above becomes

$$N_\lambda = \lfloor (82 - (-30))/0.2 \rfloor + 1 = 561 \tag{6.7}$$

$$N_\phi = \lfloor (88 - 45)/0.1 \rfloor + 1 = 431 \tag{6.8}$$

where the braces denote the "floor" function. Likewise, the mapping from projection grind indices to longitude, latitude positions become

$$\lambda[i] = -30.0 + 0.2 \times i \qquad \forall i \in \{1, 2, \ldots, N_\lambda\} \tag{6.9}$$

$$\phi[j] = 45.0 + 0.1 \times j \qquad \forall j \in \{1, 2, \ldots, N_\phi\} \tag{6.10}$$

The "regular" projection option will interpolate the model values to the two-dimensional grid defined by the indexes i and j, and their geographical positions will be positions $(\lambda[i], \phi[j])$. The grid will have the size $N_\lambda \times N_\phi$.

"native" projection option: With this option there is no horizontal interpolation, as the model values will be kept on the original (native) model grid. You can, however extract subsets of the model grid. For this projection option the file **proj.in** looks like this

```
1: native
2:     1              # first grid index in 1st grid dimension
3:   200              # last  grid index in 1st grid dimension
4:     1              # 1st grid dimension  skip
5:     1              # first grid index in 2nd grid dimension
6:   100              # last  grid index in 2nd grid dimension
7:     1              # 1st grid dimension  skip
```

This file is very similar in form to the "regular" option. The first line denotes that the "native" projection method is used. Lines 2-4 denote the first grid index to use, the last grid index to use and the spacing between the grid indexes, all for the first model grid dimension. The second grid dimension values are given in a similar manner on lines 5-7.

The number grid points in the first and second grid dimension for the example above becomes

$$N_{\mathrm{x}} = \lfloor (200-1)/1 \rfloor + 1 = 200 \tag{6.11}$$
$$N_{\mathrm{y}} = \lfloor (100-1)/1 \rfloor + 1 = 100 \tag{6.12}$$

where the braces denote the "floor" function. The resulting "subgrid" will have the size $N_{\mathrm{i}} \times N_{\mathrm{j}}$.

Note that by setting the last grid index to -1, the size of the model grids are used in stead (e.g idm or jdm).

"polar_stereographic" projection option: With this option the stereographic projecion is enabled. For this projection option the file **proj.in** looks like this (line numbers added)

```
 1: polar_stereographic
 2: -3800.             # First x-point   [~km]
 3:  3800.             # Last  x-point   [~km]
 4:  12.5              # X-point spacing [~km]
 5: -5500.             # First y-point   [~km]
 6:  5500.             # Last  y-point   [~km]
 7:  12.5              # Y-point spacing [~km]
 8: -45.0              # Central longitude
 9:  90.0              # Central latitude
10:  0.01              # Conversion factor
11:  F                 # Rotate to output grid
```

The first line denotes that the "polar_stereographic" projection method is used. Lines 2-4 denote the first x-projection point to use, the last x-projection point to use and the spacing between the x-points. The y projection values are given in a similar manner on lines 5-7.

The number of grid points in the first and second grid dimension for the example above becomes

$$N_{\mathrm{x}} = \lfloor (3800-(-3800))/12.5 \rfloor + 1 = 609 \tag{6.13}$$
$$N_{\mathrm{y}} = \lfloor (5500-(-5500))/12.5 \rfloor + 1 = 881 \tag{6.14}$$

where the braces denote the "floor" function. The mapping from projection grid indices to x, y projection coordinates become

$$x[i] = -3800.0 + 12.5 \times i \qquad \forall i \in \{1, 2, \ldots, N_{\mathrm{x}}\} \tag{6.15}$$
$$y[j] = -4500.0 + 12.5 \times j \qquad \forall j \in \{1, 2, \ldots, N_{\mathrm{y}}\} \tag{6.16}$$

The resulting grid will have the size $N_{\mathrm{x}} \times N_{\mathrm{y}}$.

Lines 8-9 denote that the center point $(x, y) = (0, 0)$ of the projection will be placed at 45° W and 90° N. The conversion factor on line 10 only affects the projection coordinate values in the resulting NetCDF file (the factor was needed for some visualization tools). The final flag on line 11 says if vectors should be

rotated to the projection grid. If set to false, vectors will be rotated to east/north directions, otherwise they will be rotated to the projection grid.

To visualize the projection when creating **proj.in**, think of the first and last (x,y) coordinate points as distances extending away from the central longitude and latitude points. As an approximation, the projection coordinate units can be considered roughly equal to kilometers. For the definition of stereographic projection, see for instance

http://mathworld.wolfram.com/StereographicProjection.html.

TODO: Clear up

"mercator" projection option: This option uses the following **proj.in**:

```
mercator
1: -100.0             # Longitude W edge
2:   20.0             # Longitude E edge
3:    400             # number of points in E-W dir
4:  -20.0             # Latitude of S edge
5:   70.0             # Latitude of N edge
6:    200             # Number of points in S-N dir
7:     0.             #Central longitude
```

The input specifies the first and last longitude values in line 1, the last longitude value, and the number of points in the longitude direction (N_x) is specified in line 3. Similarly the latitude values are given in lines 4-6. Note that it is the size of the grid which is given on line 3 and 6, *not* the grid increment which is the case of the other projections. The grid is constructed in "Mercator"-space by using uniform spacing in the projection coordinates. The final line consists of the central longitude of the projection, which defines the location of $x = 0$ in the projection.

The first and last longitude ($\lambda[1]$ and $\lambda[N_\mathrm{x}]$) values and the first and last latitude values ($\phi[1]$ and $\phi[N_\mathrm{y}]$) are given in the input files, as well as the size of the projection grid through N_x and N_y. Let $f(\lambda[i])$ and $g(\phi[j])$ be the Mercator transformations from longitude, latitude values to projection coordinates. The grid is constructed like this

$$x[i] = f(\lambda[1]) + \frac{f(\lambda[N_\mathrm{x}]) - f(\lambda[1])}{N_\mathrm{x} - 1} \times (i - 1) \tag{6.17}$$

$$y[i] = g(\phi[1]) + \frac{g(\phi[N_\mathrm{x}]) - g(\phi[1])}{N_\mathrm{y} - 1} \times (i - 1) \tag{6.18}$$

and these grid positions can be retrieved through the inverse functions $f^{-1}(x[i])$, $g^{-1}(y[j])$. These are the locations where the projection routines will interpolate the model. For the definition of the Mercator transformation itself, see for instance

http://mathworld.wolfram.com/MercatorProjection.html

6.5.3 Stations file "stations.in"

The stations file is used with the routine **hyc2stations** to set up station positions to interpolate the model to, and join these into "groups" which are logically connected, such as ocean cross-sections. The file **stations.in** will specify these points as well as names of the groups. The file typically looks like this

```
#FRAM_STRAIT
-20.00   79.00
-19.37   79.04
-18.74   79.07
-18.10   79.11
-17.46   79.14
-16.82   79.17
-16.17   79.20
-15.52   79.23
-14.87   79.25
```

```
-14.21   79.28
-13.55   79.30
     ....
 14.61   79.02
 15.00   79.00
#DENMARK_STRAIT
-37.00   66.10
-36.69   66.11
-36.39   66.12
-36.08   66.13
-35.77   66.14
-35.46   66.15
-35.15   66.16
-34.84   66.17
     ....
-22.80   66.01
-22.68   66.00
```

where most of the stations have been left out for brewity. The format is as follows. Each group must start with a name, and this is indicated by the "#" in the first column of **stations.in**. In the example there are two groups; "FRAM_STRAIT" and "DENMARK_STRAIT". After the name, follows the stations for that particular group, these stations are read until the end of the file, or until the next group name.

Using the group logic, it is relatively simple to link together station positions into sections. The tool **setupstations.py** (see Section 6.4.16) can be used for this purpose by extracting station positions along rhumb lines.

6.5.4 Depthlevels file "depthlevels.in"

The **depthlevels.in** file is a simple file which sets up the vertical depth levels the model should be interpolated to. Its format is the number of depth levels on the first line, and on the following lines are the depthlevels themselves. Like in this example:

```
12                  # Number of z levels
5.0
30.0
50.0
100.0
200.0
400.0
700.0
1000.0
1500.0
2000.0
2500.0
3000.0
```

6.5.5 Section file "sections.in"

This file is used by the routines **m2section**, **m2transport**, and **m2transport2**, to set up the grid points which are used to plot sections and calculate transports. The contents of the file **sections.in** typically looks like this(line numbers added)

```
 1:1.2 F               # File version and verbosity flag
 2: #####
 3: ##### Sections:  lon,lat pairs for start and end point.
 4: FRAM_STRAIT
 5: -20. 79.
```

```
 6:   0. 79.
 7: 15.  79.
 8: #
 9: DENMARK_STRAIT
10: -37.  66.1
11: -22.68 66.
```

The first four lines are file version flags and a header. The remaining lines denotes the section definitions. The sections are given by first, a descriptive name, then points defining the section, and finally a "#" sign in the first coloumn to denote the end of a section specification.

As an example, the section named "FRAM_STRAIT" will consist of a section segment from $(-20.0, 79.0)$ to $(0.0, 79.0)$, and then a new section segment from $(0.0, 79.0)$ to $(15.0, 79.0)$, all given in longitude/latitude pairs. These section segments will be joined together when calculating the final section for "FRAM_STRAIT". It is possible to specify more segments, with a maximum hardcoded value of 50 sections.

A sample **sections.in** file can be found in the directory **Input/**.

6.5.6 Transport file "transport.in" and scalar transport file "scalartransport.in"

Transport file transport.in The transport specification file **transport.in** is used by the routine **m2transport2** in Section 6.4.34. The file depends on the **sections.in** file as well.

In file **transport.in**, more detailed transport specifications can be set up. The transports are given names, they are then linked to the sections defined in **sections.in**. Finally the transport can be applied to specific water masses, by defining thresholds for the water masses involved in the transports. For instance, it is possible to calculate the transport, involving only water masses with a salinity larger than 35 psu. An example of **transport.in** is given here

```
T1                  FRAM_STRAIT          SALINITY     +    -999.99     34.9
T2                  FRAM_STRAIT          SALINITY     -    -999.99     34.9
T3                  FRAM_STRAIT          SALINITY     +      34.9     999.99
T4                  FRAM_STRAIT          SALINITY     -      34.9     999.99
T5                  DENMARK_STRAIT       DENSITY      +    -999.99     27.8
T6                  DENMARK_STRAIT       DENSITY      -    -999.99     27.8
T7                  DENMARK_STRAIT       DENSITY      +      27.80    999.99
T8                  DENMARK_STRAIT       DENSITY      -      27.80    999.99
T9                  DENMARK_STRAIT       MULTIPLE
T9                  DENMARK_STRAIT       DENSITY      n      27.80    999.99
T9                  DENMARK_STRAIT       TEMPERATURE  n       5.00    999.99
T9                  DENMARK_STRAIT       END
```

The first column is the unique name of the transport. The second column is the name of the section to use for this transport, this name must be defined in **sections.in**,described in Section 6.5.5.

The remaining columns sets the threshold criteria for water masses used in the transport calculations. The third column describes the variable to use for threshold criteria, it can be "SALINITY", "TEMPERATURE", "DENSITY" or "DEPTH". The values in column five and six are the lower and upper values of the variable, any water mass with a value between these two estimates are included in the transport estimates. As an example, for the transport named "T1" above, only waters with salinity larger than -999.99 (i.e. all waters) and less than 34.9 is to be included in the transport estimates.

The fourth column denotes direction, where "+" is positive transport only, "-" is negative transport only, and "n" is net transport. The positive directions of a section can be found by the following rule. When standing at the first point of a section, and looking towards the next point of the section, then the positive direction is towards your right-hand side.

For T9 multiple criteria are used. The list should start with a line for wich the criteria variable are MULTIPLE. The following lines consists of each criteria. You must end the criteria list with END.

Scalar transport - "scalartransport.in" This file can be used to make the routine **m2transport2** calculate transport for additional tracer fields, such as for nutrients in biological models. The format is

```
'temp     '  6. 3987000
'saln     '  0. 1.
```

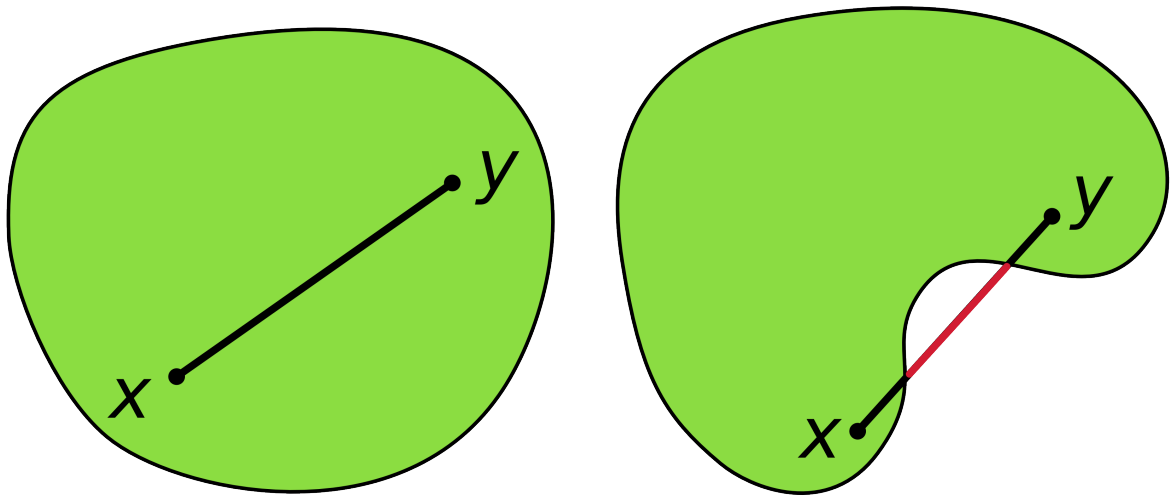
Here the first column denotes the variable to calculate the scalar transport for. The second column is an offset value which is subtracted from the variable when the transport is calculated. Finally, the third column is a scale factor, which is multiplied with the final scalar transport estimate. The example on the first line above will calculate temperature transport relative to 6 degrees Celsius, with a scale factor of $3987000 \approx c_p \times \rho$, where c_p is the heat capacity of sea water and ρ is water density. Line 1 would therefore give a heat transport estimate relative to 6 degrees celsius.

6.5.7 The regiondefs file "regiondefs.in"

This file is used to calculate regional averages from the model. It allows you to identify regions by their names and by the vertices outlining outlining the region. An example is shown below(line numbers added)

```
 1: 2                                  # Number of regions
 2: 4 TEST1                            # Number of points and name of 1st region
 3: -30 40                             # lon/lat pair - 1st point
 4:  30 40                             # lon/lat pair - 2nd point
 5:  30 60                             # lon/lat pair - 3rd point
 6: -30 60                             # lon/lat pair - 4th point
 7: 4 TEST2                            # Number of points ansd name of 2nd region
 8: -30 20                             # etc etc ...
 9:  30 20
10:  30 60
11: -30 60
```

The first line gives the number of regions to read from the file. On the next line is the number of vertices (N) for the region, followed by the unique name of the region. Lines 3-6 then contain the N positions of the vertices. Lines 7-11 contain another region definition and so on.



(a) Convex region (b) Non-convex region

Figure 6.3: Examples of convex and non-convex regions.

Two things are important when defining a region. First of all the region most be convex. Roughly speaking this means that every straight line between two points inside the region must be contained entirely inside the region, see Figure 6.3. Secondly, the path around the region formed when specifying must be traversed either clockwise or anticlockwise. These are limitations which should be fairly easy to circumvent. For instance by splitting a non-convex region into two convex regions.

6.5.8 The NCAR graphics input files - “cplot.in” and “cvplot.in”

6.6 Matlab tools

6.6.1 abfile

The matlab tool “abfile” is a object-oriented approach to reading data from hycom files into matlab. The tool consists of one file, **abfile.m**, and is presently able to read several types of input files. The file type you want to read must be given as an input argument to the abfile constructor method. The abfile toolbox is documented in matlab. Type “help abfile” for some info. You can also get information on abfile methods by typing “help abfile.abfile” or “help abfile.getfield” for instance. To see a list of abfile methods, type “methods(abfile)” in matlab. For a brief explanation and examples of usage, read the following paragraphs.

Initializing file objects To use the abfile tool to read hycom data files, you must first create a abfile object, this is done by calling the abfile constructor. A typical call looks like this in matlab

```
obj=abfile('F00restart2007_330_00.a','restart')
```

This initializes the abfile object “obj”, by reading header data for the file **F00restart2007_330_00.a**, and by saying that this file is a restart file in the second argument. Possible file types are “restart”, “archv”, “nersc_daily”, “nersc_weekly”, “forcing”, “regional_grid”, “relax” and “raw”. The last file type is for a file which doesnt fit any of the other file types. In this case data is read “blindly”, without any knowledge of the names of the fields in the file.

Reading field data Once the abfile object is created, you can start working with it. Typical methods are reading fields for a given model layer, a given time step, and for a given variable name. There are also methods for inquiring the contents of the file. The following statement will read the temperature variable for time step 1 and vertical level 1 into the variable “fld”

```
fld=obj.getfield('temp',1,1 );
```
[1]

Here the first argument “temp” is the variable to extract from the restart file, the second variable is the vertical depth level to extract, and the third argument is the time step to extract. It is also possible to pass empty data to the routine to retrieve several depth levels or several time levels in one go. The following will read all vertical depth levels for time step 1

```
fld=obj.getfield('temp',[],1 );
```

Be careful with this approach, however, as it is an excellent way of abusing the memory of the machine you work on.

Reading point data In addition to reading field data, it is also possible to read point data from a file. This is done with the method “getpoint”, an example of this routine is

```
pnt=obj.getpoint(100,100,'temp',[],1);
```

which will read all vertical levels of the temperature variable, at time step 1, and at grid position $(100, 100)$ into the variable “pnt”. You can also supply several points to this routine by specifying vectors for the points:

```
pnt=obj.getpoint([100 100],[100 101],'temp',[],1);
```

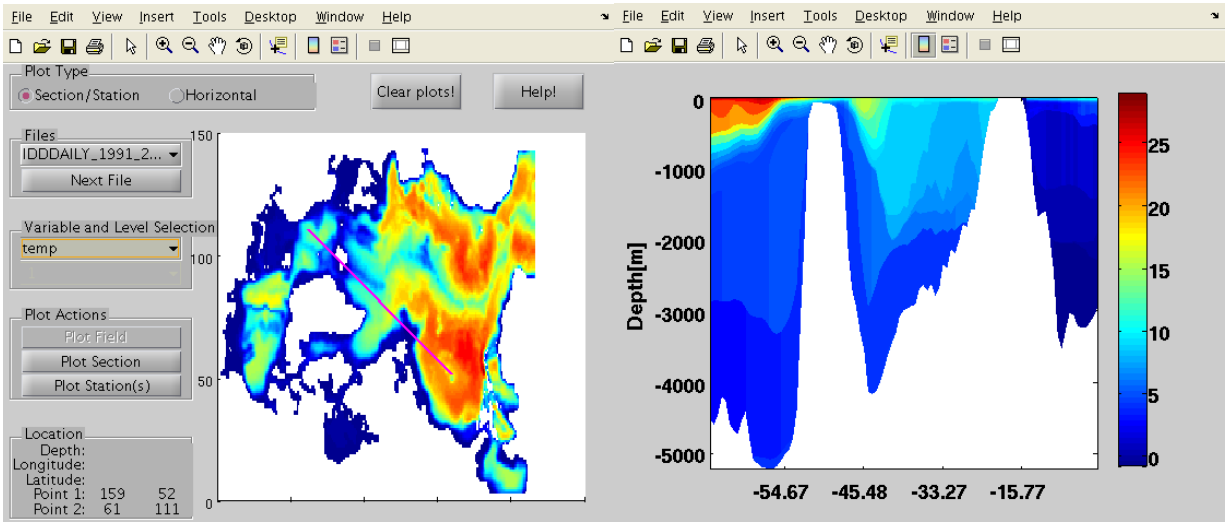
This will read the data at grid positions $(100, 100)$ and $(100, 101)$.

Inquiring on the contents of a file There are also methods by which you can inquire the contents of a file. The methods can inquire on the variables, layers and time steps present in a file. To inquire for variable names use the “getfieldnames” method

```
obj.getfieldnames()
```

which returns the names of variables in the file, these names can then be passed to the getfield and getpoint

[1] Note that an equivalent way of doing this is with the statement “fld=getfield(obj,'temp',1,1);”. Use the approach you prefer.



(a) hycomvis GUI

(b) hycomvis section plot of temperature for the section selected in Figure 6.4a

Figure 6.4: The hycomvis GUI and example plots

methods. To inquire on the vertical layer indexes use the following methods

obj.getlevels()

obj.getlevels('temp')

The first method will retrieve all possible layer indexes for the file as a whole, while the second method will retrieve all possible layer indexes for the temperature variable. Note that 2D variables usually return 0 as layer index. Similarly you can inquire for time levels in the file the following methods

obj.gettimes()

obj.gettimes('temp')

Some files have just one time index (ex "nersc_daily") while others can have several (ex "restart").

6.6.2 Hycomvis

The hycomvis tool can be used to visualize the contents of a hycom file. It uses the "abfile", Section 6.6.1 tool to read the contents of the hycom files. The tool is started from the matlab command-line by specifying the file(s) to process and the file types. The file types are the same as those given to the abfile tool. The following command shows how to start hycomvis with two "nersc_daily" files, and one "nersc_daily" file

hycomvis({'IDDDAILY_1991_217_1992_073.a', 'IDDDAILY_1991_217_1992_074.a'},'nersc_daily');

hycomvis('IDDDAILY_1991_217_1992_073.a','nersc_daily');

Note that the "nersc_daily" files are in the first example passed to the hycomvis routine as a cellstring array, denoted by the curly braces. In the second case only a single file is passed to hycomvis as a character string. It is also possible to pass a character array to the routine - you can for instance pass the output of the "ls" function in matlab to the hycomvis routine :

files=ls('IDDDAILY*.a');

hycomvis(files,'nersc_daily');

Once hycomvis starts, you should be presented with the GUI shown inf Figure 6.4a.

The GUI consists of a selection map, buttons and menus. By clicking and dragging in the selection map (the bathymetry plot), you can select regions, cross-sections or stations, depending on the plotting mode you are in. The hycomvis tool has two plotting modes, given by the buttons in the upper left-hand side. It can be plotting either in "Section" mode or in "Horizontal" model. When "Section" mode is enabled, clicking will give a selection as a point indicated by a magenta "+" sign. By click-and-drag you will select a section, indicated by a magenta line. In this mode you can create Section plots or station plots,

indicated by the active buttons in the "Plot Actions" button group. An example of a section plot is shown in Figure 6.4b. When Horizontal mode is enabled, click-and-drag operations in the bathymetry map will create rectangular areas. In this mode you can create horizontal field plots, indicated by the active button in the "Plot Actions" button group.

In both modes you can select the active file in the Files menu, or go to the next file in the file menu by clicking "Next File" button. You can also select the variable to display in the "Variable and level selection" menu. In "Horizontal" plot mode you can also select the layer index to plot. It is possible to have several plots of different types open at the same. A feature of the hycomvis tool is that the plots created are updated as the file changes. When you select a new file in the Files menu, or click "Next File", any plots you have created will be updated with the contents of the new file. To close the plots created by hycomvis, click the "Clear Plots" button. This will not close the main hycomvis window.

The hycomvis tool has no methods for changing the properties of the plots which are created . Some features of the plots are kept as the plots are updated, however:

- Section plots keep the color axis (caxis), X limits (the 'XLim' axes property), and Y Limits (the 'YLim' axes property).
- Horizontal plots keep the color axis(caxis).
- Station plots keep the X limits(the 'XLim' axes property) and Y Limits(the 'YLim' axes property).

This means that you can change these properties of the plots created by hycomvis, and any plot updates should keep these changes.

6.6.3 Topofix

Topofix is a small matlab GUI for modifying a depth matrix. It consists of many of the bethymetry modification tasks that you would typically do with the conformal mapping tools B. The GUI consists of a bathymetry map, and several buttons doing various tasks. In the map you can select points with mouse clicks, and regions with click-and drag operations. You can start the topofix utility in two ways, either by specifying the size of the model grid;

```
topofix('size',[216,144]);
```

In this case topofix will read the grid from any **regional.depth.a** file you have in the working directory. Another way to start it is by specifying the depths matrix as an input

```
topofix('depths',depth);
```

where "depth" is a matlab variable. This can be handy if you want to do some advanced operations in matlab which you can't do in topofix, such as bathymetry smoothing, for instance.

When topofix starts successfully, you should have a GUI similar to Figure 6.5. In the bathymethry view, click and drag operations will select regions in the map view. Figure 6.5 shows a selected region as a rectangular box. The buttons on the left-hand side do various tasks:

Reset View Click this button if you want to reset the view for any reason.

Toggle Grid Switches the grid lines on and off.

Change color axis By modifying the min and max values directly below this box, and clicking this box the min and max colormap values will change.

Check isolated Checks for points with more than three land neighbours and closes them.

Workspace dump Puts the current depth matrix in the matlab workspace. This way you can modify them "by hand" in matlab.

File dump Saves the bathymetry to hycom depth files. The names are prefixed with "topofix.", so you wont overwrite existing **regional.depth.[ab]** files

Reset depths Resets the depths to what they were when you started topofix.

Undo last Undoes the last bathymetry change operation.

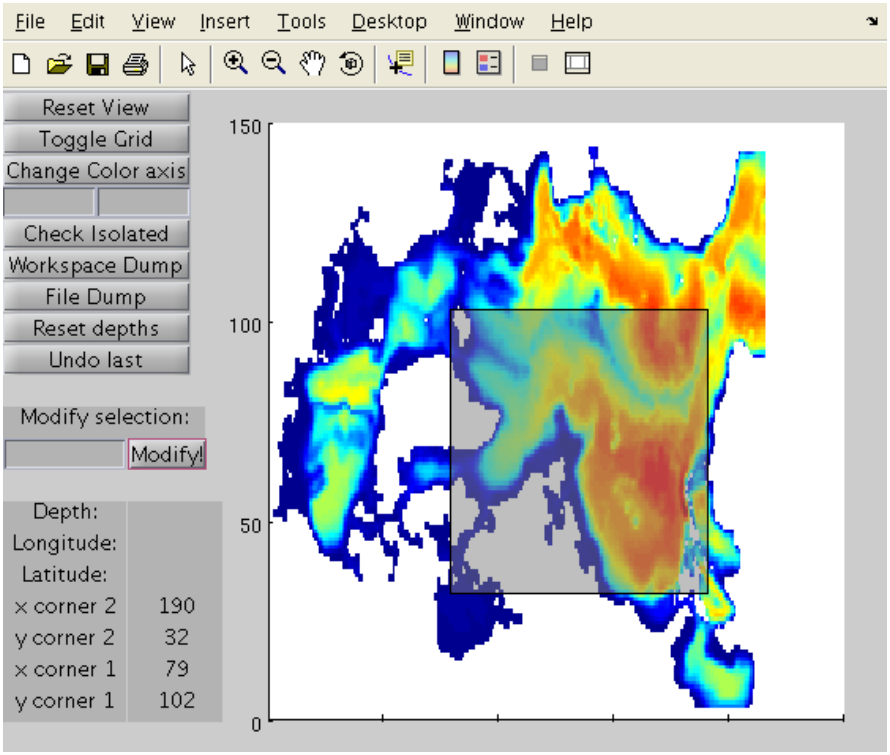


Figure 6.5: The topofix GUI

Modify selection By typing a value in the text box below, you can modify the depths of the current selection in the map.

In addition to this you can probe for values by clicking on a point in the map. The longitude, latitude and depth values will then show up as values in the information on the lower left-hand side. It is also possible to zoom in on a region in the map by using the zoom button on the toolbar of the figure, just remember to disable the zoom button before you start modifying bathymetry values again. If you are unable to zoom out again, you can press the "Reset View" button.

Note that there are no buttons for modifying the east/west, south/north boundaries, this should be done by the conformal mapping routines, section B.

Appendix A

Checklists for setting up a model

A.1 Standard setup

This is a pointwise run-through of the steps needed to set up and run NERSC-HYCOM.

A.1.1 Step 0. Retrieve code and compile

You need to retrieve the code from the svn as explained in the introduction, and compile the MSCPROGS (and HYCOM_ALL if you are the first one on the machine) utilities as explained in Section 6

A.1.2 Step 1. Configure region

Skip this step if a region is already set up. More details can be found in section 3.1.

- Decide on a suitable region name. Suggested naming scheme is "XXXYZ.ZZ", where XXX is a three letter ID which should be somewhat descriptive of the region. Y is a lowercase letter (a, b, c, etc) indicating minor revisions to a model grid. Z.ZZ is an indicator of resolution, example 0.10 for a grid with 1/10 degree resolution
- Set up new region directory. This is best done by running the script **bin/newregion.sh** under an already existing region directory. A "template" region directory is copied with the svn, under

  ./hycom/HYCOM_2.2.37/NATa1.00
- The **newRegion.sh** script can be run without arguments for quick info on its usage. When it is finished check the file **REGION.src** to make sure the variable "R" is set to the region name.
- Copy the **regional.grid.[ab]** files, as well as the file **grid.info** into the **topo/** subdirectory of the region directory. These files must be created with the conformal mapping tool. Do not copy topography files at this stage.

A.1.3 Configure topographies

After setting up a region, the next step is to set up a topography. Topographies are usually imported from an old model run or modified locally (mainly for nesting). See section 3.2 for more info.

- Decide on a topography version number. Typically you should start with 00 (the one produced from grid) and work your way upwards.
- Create or import topographies. You can either import an existing topography or modify an existing one (for nesting):
  - Import an existing topography file. Topography files from old NERSC-HYCOM model runs have a different land value than in version 2.2, so they need to be modified in order to use them in v 2.2. The script **topo/bin/regioncopy.sh** should take care of that. Run with no arguments to get info on its usage.

- Or use the **topo/bin/nestbat.sh** script to generate a topography for a nested model. Also needs information on the model to nest from.
- Make sure that the topography is robust, that they are no island or too abrupt change in the 3 first grid cells

A.1.4 Configure experiment

Skip this step if a experiment is already set up. More details can be found in section 3.3.

- Decide on a experiment number - it should be a three-digit floating point number with one decimal digit. Examples are $01.0, 01.1, 01.2, ..., 99.8, 99.9$.
- The region setup process in step 1 should by default set up a experiment directory for experiment 01.0. The name of this default experiment directory is "expt_01.0". You may use that as a starting point. If you need to create a new experiment directory based on an already existing experiment, the script **bin/newExpt.sh** can do this for you.
- Make sure the file **EXPT.src** in the experiment directory reflects the experiment number (set in variable "X") and that the variable "E" is the same as "X", but without the period. As an example, if X=01.0, we would have E=010.
- Modify the variable "T" in **EXPT.src** to link this experiment to one of the topography versions in the **topo/** subdirectory.
- You must also set up the **blkdat.input** properly. As a minimum make sure the variables "iexpt", "idm", "jdm" are set properly as weel as information about the vertical structure. "iexpt" must match the variable E in **EXPT.src**, "idm" and "jdm" must match those of **regional.grid.b**. For the vertical coordinate you need to chose the number of layers, the reference densities, the number of hybrid layer ...

A.1.5 Configure relaxation data files

This step sets up the data files needed to initilize the model, and for running with climatology relaxation. See sections 3.4 for more info.

- Look at the **blkdat.input** and get the value of the thflag parameter.
- Decide on whether to use PHC or Levitus climatologies (supplied with standard HYCOM), and call scripts **relax/bin/z_phc.sh** or **relax/bin/z_levitus.sh** with the thref value as parameter. The PHC climatology is recommended.
- Run the script **relax/bin/relax.sh** to generate ocean climatologies for HYCOM.
- Run the script **relax/bin/relax_rmu.sh** to generate a relaxation mask. You will be asked some questions as the script runs.
- If you use the "old" climatology when setting up forcing, you will need to run the scipt **relax/bin/old_lev_nersc** as well.

The cimatologies are now located in **relax/$E/** where $E corresponds to the value of E in **EXPT.src**.

A.1.6 Configure forcing files

This step sets up the forcing files needed to run the model, which includes atmospheric and river data sets, see Section 3.5.

- To generate climatology forcing files, call the script **force/bin/nersc_clim.sh**. It is possible to generate climatologies from several data sets (er40, ncep etc) and keep them in the **force/** subdirectory. This is usually needed when running with climatology forcing, but also when using the "INLINE_FORCING" CPP option in NERSC-HYCOM, see section 4.1.

- To generate synoptic forcing files, call the script **force/bin/nersc_synoptic.sh**. The file **infile.in** in the experiment directory must be set up before running this script. Only needed if running with the forcing flag "prep" set in **infile.in**.
- To generate river forcing files, call the script **force/bin/nersc_trip.sh** or **force/bin/nersc_rivers.sh**. The latter relies on the existence of a file **rivers.dat** in the force directory while the former will compute an estimate from an hydrological model. Only needed when running with river forcing switched on in **blkdat.input**.
- To generate seawifs water type forcing files, call the script **force/bin/seawifs_mon_kpar.sh**. This is only needed if running HYCOM with kpar=1, set in **blkdat.input**.

The forcing files should now be set up in the different subdirecories of **force/**.

A.1.7 Configure MPI and compile

This step sets up the MPI partitioning and compiler flags needed to run the model, Section 3.3.4 and section 4.2.

- Set up tile partition files using the script **topo/bin/tile_grid.sh**. This will create files in the **topo/partit/** directory. The last number in this file name is the number of MPI tasks that will be used.
- Link (or copy) the **src_2.2.12/** and **config/** files into the region directory, they can be found in this location on hexagon

  ./hycom/HYCOM_2.2.37/CodeOnly/

  Remember to keep the same directory names.
- Copy the the build directory into the region directory, but rename it to **Build_V2.2.12_X01.0/**, where the last number is the experiment number (here 01.0 is the experiment number).
- Run the script **setuppatch.sh** in the build directory. Input is the number of MPI tasks to use. This number must match the number of MPI tasks in an already existing patch-file in the **topo/partit/** directory.
- Set up makefile configuration files, libraries and CPP flags in the file **flags** in the build directory.
- Compile the model with "make"

You should now have the executable in the build directory.

A.1.8 Configure run-time options and run

This step sets up the final setup of the model forcing files needed to run the model, which includes atmospheric and river data sets, see Section 5.

- Make sure that the data and scratch directories are properly set in **EXPT.src** in the experiment directory.
- Edit the run-time configuration files **blkdat.input**, **infile.in**, etc to make sure they are properly set up.
- Remember that to initialize from climatology, you must create an empty file called "INIITIALIZE" in the experiment directory. The same must be done when copying a restart file to another date.
- Run the script **preprocess.sh** to make sure you have all the files necessary to run the model. If something is missing, you will be told. Make sure **preprocess.sh** runs without errors before running the model.
- Edit the job script, set up the number of MPI and OpenMP tasks (if they are not automatically set by the script).
- Finally run the model by submitting the job script.

A.2 Other setup options

The standard setup, Section A.1, is the bare minimum needed to run the model. In order to run the model with more options, the following additional steps are needed.

A.2.1 Nesting (outer)

To set up outer nesting some additional steps are also needed. here, see Section 3.2, Section 3.6, and Section 5.2.1 for additional info.

- Run the script **nest_nersc/bin/nest_outer.sh**. This will set up interpolation info needed when interpolating from the outer model to the inner model. This step can be done at any time after the topography is set up in the current model and the model which will receive nesting data.
- Before starting the model make sure that the path in the file **nesting.in** has written permission.
- Switch on outer nesting in **infile.in**

A.2.2 Nesting (inner)

To set up inner nesting some additional steps are needed which will be given here, see Section 3.2, section 3.6 and Section 5.2.2 for additional info.

- In the topography checklist, given in Section A.1.3, the **nestbat.sh** script should be used. This will smooth the model topography towards the outer model. The newly created topography should also be used by setting topography version in **EXPT.src** in the experiment directory.
- Run the script **nest_nersc/bin/nest_inner.sh**. This will set up nesting relaxation masks and ports needed when running the model. This step can be done at any time after the nestbat step.
- Before starting the model, the nesting conditions must be linked or copied into the current experiment directory as

  ./nest_R_T/

  The values of R and T must match the region name and topography version of the model in this experiment. Note that this is the same name that the outer model will save nesting conditions to, see Section A.2.1
- Switch on inner nesting in **infile.in**

A.2.3 Tidal forcing

See Section 3.7 for additional info.

- Call one of the scripts **tides_nersc/bin/tide_CSR.sh** or **tides_nersc/bin/tide_FES.sh**.
- Switch on tides in **infile.in**. Make sure the tidal data set matches what was set up in step 1 (FES or CSR).

Appendix B

Conformal mapping tool

The HYCOM model is defined on a two-dimensional lattice of points, with each point describing a scalar variable on this lattice given by an index pair (i, j). This point is usually referred to as the p-point. The grid is a so-called Arakawa C-grid where u and v velocity points are offset from the p-point. The first velocity component (u) is placed on the midpoint between two neighbouring p-points $(i-1, j)$ and (i, j) The second velocity component (v) is placed on the midpoint between two neighbouring p-points $(i, j-1)$ and (i, j). See **??** for an illustration.

In HYCOM the actual location of the grid lattice on the earth is described by the variables plon,plat (p-points), ulon,ulat (u-points) and vlon, vlat (v-points), given in the file regional.grid.[ab]. We will loosely refer to the lattice of grid points (i, j) and the information of the geographical positions of u, v, and p-points as the "model grid".

The model grid is a unique feature of the region setup. For different experiments you may change the topography, the forcing, code features etc, but the grid files (**regional.grid.[ab]**) should not change. These files determine the longitude and latitude of grid points, as well as distances between grid nodes. The files related to the model grid should be placed in the **topo/** subdirectory. This includes the files **regional.grid.[ab]**, the conformal mapping file *grid.info*, and also old grid files such as **latlon.dat** and **newpos.uf**. The two latter files are rarely needed anymore, but its ok to keep them in the **topo/** directory just in case.

The creation of a model grid at NERSC is usually done with the conformal mapping routine "grid", which can be found in the directory *$MSCPROGS/bin_setup/*. It is also possible to set up a new grid using tools supplied in standard HYCOM. Routines for this can be found under the directory *$HYCOM_ALL/topo/*. Many of the tools found under *$MSCPROGS/* depend on the grid being a conformal mapping grid, however, so to use these it is safest to use the conformal mapping grids.

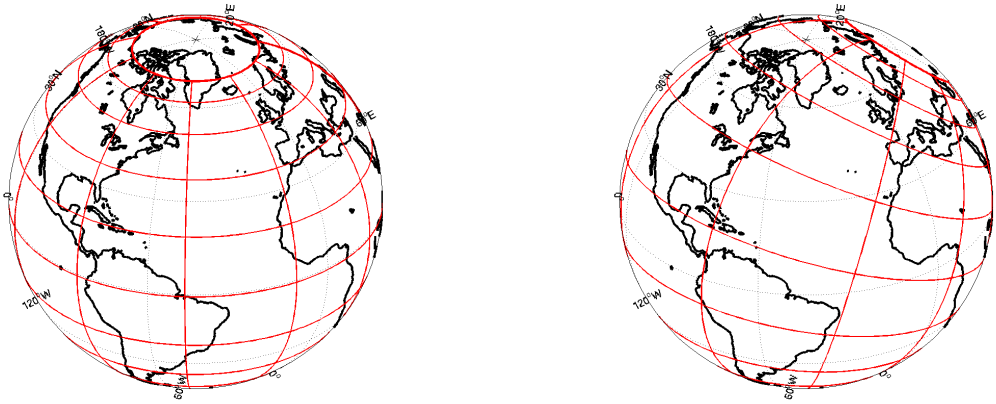
B.1 Input files

B.2 Model Grid generation with the conformal mapping tools

The conformal mapping routines (**?**) defines the mapping from the two-dimensional lattice of points given by grid indexes (i, j) to their geographical location in longitude and latitude coordinates. This makes it possible to place a model grid in a region with great flexibility, or to create global grids which avoid the singularities at the poles. It also makes it possible to shape the grids without too much distortion (variation of grid size as one moves from one grid cell to the next). This is important since too much distortion can affect the effective precision of numerical schemes.

The conformal mapping tools rely on a input file called grid.info. This file determines the location of the model grid, and can also be used to set the bathymetry. Its contents look somewhat like this (line numbers added for reference in the text)

```
line 1 :    -40.0 140.0     ! Position of pole N (lattitude, longitude):
line 2 :    -50.0 140.0     ! Position of pole S (lattitude, longitude):
line 3 :    177.5 182.7 200 ! Longitude interval West lon,   East lon, idim
line 4 :      3.0  80.0 220 ! Lattitude interval south limit, north limit, jdim
line 5 :   .false.          ! Generate topography
```



(a) New poles of the conformal mapping co-located with actual north and south poles

(b) New North Pole of the conformal mapping shifted towards Siberia

Figure B.1: The effect of moving the new poles of the conformal mapping routine

```
line 6 :   .true.           ! dump teclatlon.dat
line 7 :   .true.           ! dump micom latlon.dat
line 8 :   .true.           ! mercator grid (true, false)
line 9 :   -2.6 .false.     ! merc fac
line 10:   .false.          ! Smoothing, Shapiro filter
line 11:   8   2            ! Order of Shapiro filter,  number of passes
```

The two first lines of this file determines the location of the north and south poles for the new grid. These can be placed anywhere on the earth - if they are placed on opposite sides of the earth, the result is a lon/lat grid in the new coordinates which are similar to standard lon/lat coordinates (though usually not co-located). An example is shown in figure B.1a where the North and South poles of the grid are placed at the geographical North and South poles. Note how the constant longitude grid lines follow great circles, as with usual longitude coordinates.

If we shift the North Pole of the grid towards Siberia, as in figure B.1b, the result is a grid with curved lines relative to great circles. By manipulating the location of the poles like this, it is possible to set up a grid with focused resolution in key areas, and lower resolution in less interesting areas. Figure B.1b also illustrates how the conformal mapping tool can be set up so that singularities of the model grid (North/South poles) are on land.

Line three and four of grid.info determine the placement of your grid and the size of the grid. While you can use the conformal mapping tools to set up a global grid, you can also set it up to give a grid covering only a part of the globe, which is useful for creating grids covering specific regions. The third line of grid.info specifies the East-West longitude interval, along with the number of grid points over this range:

```
line 3 :    177.5 182.7 200 ! Longitude interval West lon,   East lon, idim
```

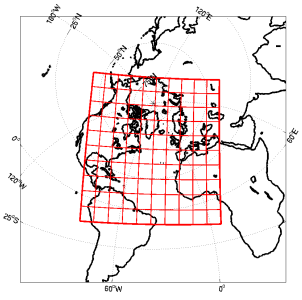
It is important to remember that the longitude interval is in the new coordinates defined by your initial placement of the new North and South Poles. By specifying a range which is less than 360 degrees, you will create a regional grid (i.e not global). Choosing the correct values for the limits depends on the placement of the poles and how far apart they are. It can be hard to visualize this so some trial and error may be required to get the correct placement. It is often a good idea to start with a wide logitude range to get the big picture.

The fourth line is the greatest source of confusion, in the example it reads:

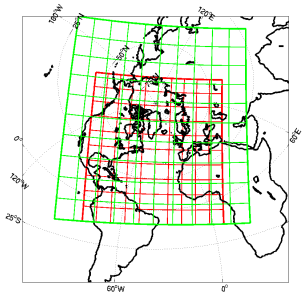
```
line 4 :      3.0  80.0 220 ! Lattitude interval south limit, north limit, jdim
```

The confusion is often related to the setting of the mercator factor, given on line 9.

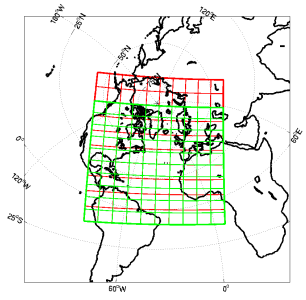
```
line 9 :   -2.6 .false.     ! merc fac
```



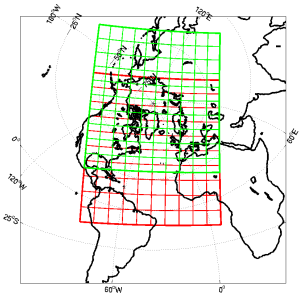
(a) Original Grid



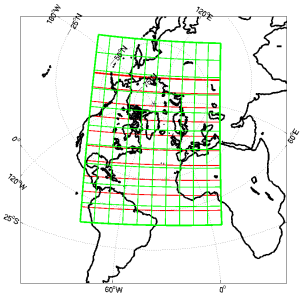
(b) Increasing east-west range



(c) Increasing grid points along east-west direction



(d) Increasing Mercator Factor



(e) Increasing grid points along north-south direction

Figure B.2: The effect of changing values in the grid.info file. Red is original grid, green is grid after modifying one set of values.

If the logical value on this line is set to .false. (which is generally recommended), the grid tool will try to create equal area grid cells. This means that the grid cell size is uniquely determined by the settings on line 3. This lays restrictions on the way the north and south limits of the grid can be set. In stead of setting the north-south range of the grid, you can now only adjust the southern limit of the grid. The southern limit of the grid is not set on line 3, however, it is set by the floating-point value on line 9 (!)

The placement of the grid will not be described in detail here, it is usually a process of trial and error. It is also hard to generalize the process. The following illustrations should hopefully be of some assistance, though. The main assumption here is that you have decided on the positions of the poles, and what remains is to move the grid to its final placement. The images of Figure B.2 will illustrate some of the effects of changing east/west limits and the merc fac southern limit.

The original grid is given in Figure B.2a. It corresponds to the grid.info file given at the start of this section. It is placed over the North Atlantic and the Arctic, and is same grid as the one used in TOPAZ3. Now suppose that some changes to the grid placement are needed. The grid can be shifted east-west by modifying the eastern and western limits, and at the same time keeping the east-west range constant. Once we start to change the east-west range or the number of points in the east-west direction, however, the grid might change in non-intuitive ways. The following describes some of the most common pitfalls. Note that grid directions refer to the new longitude and latitude coordinates, not actual longitude and latitude,

Inreasing the east-west range When the east-west range is changed, this will also change the size of grid cells. As the grid length is set to be the same in the east-west and north-south direction, this means that a change in east-west range will affect the north-south range of your grid. Figure B.2b illustrates this. In this case the east-west range is increased. As the east-west range is increased the grid cells will increase in size, both in east-west and north-south direction. This means that the northern limit of the grid will increase as well. This is seen in Figure B.2b where the increased east-west range has lead to a increased coverage in both the east-west and north-south directions. (Red=original grid, Green=changed grid). The opposite effect is seen when *reducing* the east-west range. Because of this behaviour, one should be careful not to increase the east-west range so much that the northern edge of the grid "hits" the north pole.

Increasing grid points in the east-west direction When number of points in the east-west direction is increased, the effect will be to decrease the size of the grid cells. Since the grid cell size in the north-south direction is decreased correspondingly, this will lead to the northern edge of the grid moving southward. This is seen in Figure B.2c. Note that the grid covers the same range in the east-west direction, while the northern limit of the new grid (in green) has moved towards the south. The opposite movement of the northern edge of the grid will happen when you reduce the grid points in the east-west direction.

Increasing the Mercator Factor When the mercator factor is increased, the new grid will shift towards the north in the new latitude direction. This is illustrated in Figure B.2d. Note that the east-west range stays the same. The opposite effect will happen when you reduce the mercator factor. As previously mentioned only the mercator factor should be modified when the grid is to be shifted in the north-south direction. One should be careful not to increase the southern limit so much that the northern edge of the grid "hits" the north pole.

Increasing the grid points in the south-north direction When you increase the grid points in the south-north direction, the size of grid cells will not change. This means that increasing the grid points leads to a northward shift in the northern edge of the grid. This is illustrated in Figure B.2e. The opposite will happen when you reduce the grid points in the south-north direction.

Adjusting resolution while keeping the same coverage To keep the coverage of the grid, while increasing/reducing the resolution, you should multiply the grid points in the east-west and north-south direction by the same factor.

Final tips

The grid tool is very flexible, but not always intuitive to use. Because of this, it is hard to give a single recipe for setting up a grid. Some general guidelines can be given, however:

- Decide on pole placements first, it can be hard to adjust them when you have "zoomed in" on a region
- Start with a wide east-west range and low resolution to begin with. This way its easier to locate your grid in matlab (for instance)
- Change one parameter at a time - changing more than one at a time can change the grid placement in spectacular ways
- Use small increments - make fequent backups of grid.info
- Use a good tool to visualize the grid. A matlab routine "showconfmap" is designed for this. It runs the "grid" routine, and plots the grid on a map projection (requires the m_map matlab toolbox)
- Switch off topography generation until you are reasonably happy with the grid placement - this significantly speeds up the grid routine.

Bibliography

Aagaard, K., and E. C. Carmack, The role of sea ice and other fresh water in the arctic circulation, *J. Geophys. Res.*, *94*, 14,485–14,498, 1989.

Anderson, J. L., An ensemble adjustment kalman filter for data assimilation, *Mon. Wea. Rev.*, *129*, 2884–2903, 2001.

Bentsen, M., G. Evensen, H. Drange, and A. D. Jenkins, Coordinate transform on a sphere using conformal mapping, *Mon. Wea. Rev.*, *127*, 2733–2740, 1999.

Bleck, R., An oceanic circulation model framed in hybrid isopycnic–cartesian coordinates, 2002.

Bleck, R., and L. Smith, A wind-driven isopycnic coordinate model of the north and equatorial atlantic ocean. 1. model development and supporting experiments, *J. Phys. Oceanogr.*, *95*, 3273–3285, 1990.

Burgers, G., P. J. van Leeuwen, and G. Evensen, Analysis scheme in the Ensemble Kalman Filter, *Mon. Wea. Rev.*, *126*, 1719–1724, 1998.

Chapman, D. C., and R. C. Beardsley, On the origin of shelf water in the Middle Atlantic Bight, *J. Phys. Oceanogr.*, *19*, 384–391, 1989.

Dickson, R. R., J. Meincke, S. A. Malmber, and A. J. Lee, The "great salinity anomaly" in th northern atlantic 1968–1982, *Progr. Oceanogr.*, *20*, 103–151, 1988.

Drange, H., and K. Simonsen, Formulation of air-sea fluxes in ESOP2 version of MICOM, *Tech. Rep. 125*, Nansen Environmental and Remote Sensing Center, Bergen, Norway, 1996.

Dümenil, L., K. Isele, H. J. Liebscher, U. Schröder, and K. Wilke, Discharge data from 50 selected rivers for GCM validation, *Tech. Rep. 100*, Max Planck Institute, 1993.

Emery, W. J., C. Fowler, and J. Maslanik, Arctic sea ice concentration from special sensor microwave imager and advanced very high resolution radiometer satellite data, *J. Geophys. Res.*, *99*, 18,329–18,342, 1994.

Evensen, G., Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics, *J. Geophys. Res.*, *99*, 10,143–10,162, 1994.

Gaspar, P., Y. Gregories, and J.-M. Lefevre, A simple eddy kinetic energy model for simulations of the oceanic vertical mixing: Tests at station Papa and long-term upper ocean study site, *J. Geophys. Res.*, *95*, 16,179–16,193, 1990.

Gill, A. E., Circulation and bottom water production in the Weddell Sea, *Deep-Sea Res.*, *20*, 301–317, 1973.

Haugen, V. E. J., and G. Evensen, Assimilation of sla and sst data into an ogcm for the indian ocean, *Ocean Dynamics*, *52*, 133–151, 2002.

Hibler, W. D., III, A dynamic thermodynamic sea ice model, *J. Geophys. Res.*, *84*, 815–846, 1979.

Holland, D. M., and A. Jenkins, Modelling thermodynamic ice-ocean interactions at the base of an ice shelf, *J. Phys. Oceanogr.*, *29*, 1787–1800, 1999.

Holland, M. M., C. M. Bitz, M. Eby, and A. J. Weaver, The role of ice–ocean interactions in the variability of the north atlantic thermohaline circulation, *J. Clim.*, *14*, 656–675, 2001.

Houtekamer, P. L., and H. L. Mitchell, Data assimilation using an ensemble Kalman filter technique, *Mon. Wea. Rev.*, *126*, 796–811, 1998.

Hunke, E. C., and J. K. Dukowicz, An elastic-viscous-plastic model for sea ice dynamics, *J. Phys. Oceanogr.*, *27*, 1849–1867, 1997.

Jazwinski, A. H., *Stochastic processes and filtering theory*, Academic Press, 1970.

Kwok, R., Sea ice concentration estimates from satellite passive microwave radiometry and openings from SAR ice motion, *Geophys. Research Letters*, *29*, 10.1029/2002GL014,787, 2002.

Large, W. C., J. C. McWilliams, and S. C. Doney, Oceanic vertical mixing: A review and a model with a nonlocal boundary layer parametrization, *Rev. Geophys.*, *32*, 363–403, 1994.

Legates, D., and C. Willmott, Mean seasonal and spatial variability in gauge–corrected global precipitation, *Int. J. Climatol.*, *10*, 110–127, 1990.

Levitus, S., and T. P. Boyer, *World Ocean Atlas 1994 Volume 4: Temperature*, NOAA Atlas NESDIS 4, Washington, D.C., 1994.

Levitus, S., R. Burgett, and T. P. Boyer, *World Ocean Atlas 1994 Volume 3: Salinity*, NOAA Atlas NESDIS 3, Washington, D.C., 1994.

Montgomery, R. B., Circulation in upper layers of southern north atlantic deduced with use of isentropic analysis, in *Papers in Phys. Oceanogr. and Meteorol.*, vol. VI, p. 55 pp., Woods Hole Oceanogr. Inst., Woods Hole, Mass., 1938.

Reichle, R. H., D. B. McLaughlin, and D. Entekhabi, Hydrologic data assimilation with the Ensemble Kalman Filter, *Mon. Wea. Rev.*, *1*, 103–114, 2002.

Rudels, B., H. J. Friedrich, and D. Quadfasel, The Arctic circumpolar boundary current, *Deep-Sea Res. II*, *46*, 1023–1062, 1999.

Semtner, J., A. J., A model for the thermodynamic growth of sea ice in numerical investigations of climate, *J. Phys. Oceanogr.*, *6*, 379–389, 1976.

Slutz, R., S. L. J. Hiscox, S. Woodruff, R. Jenne, D. Joseph, P. Steurer, and J. Elms, *Comprehensive Ocean-Atmosphere Data Data Set; Release 1,NTIS PB86-105723*, NOAA Environmental Research Laboratories, Climate Research Program, Boulder, Col., 1985.

Svendsen, E., K. Kloster, B. Farrelly, O. M. Johannessen, J. A. Johannessen, W. J. Campbell, P. Gloersen, D. Cavalieri, and C. Mätzler, Norwegian remote sensing experiment: Evaluation of the nimbus 7 scanning multichannel microwave radiometer for sea ice research, *J. Geophys. Res.*, *88*, 2781–2791, 1983.

Thorndike, A. S., and R. Colony, Sea ice motion in response to geostrophic winds, *J. Geophys. Res.*, *87*, 5845–5852, 1982.