Naval Research Laboratory

Stennis Space Center, MS 39529-5004

NRL/MR/7320--08-9154

User's Manual for the Polar Ice Prediction System (PIPS) Version 3.0

PREPARED FOR:

Naval Oceanographic Office
Systems Integration Division

PREPARED BY:

PAMELA G. POSEY
LUCY F. SMEDSTAD
RUTH H. PRELLER
E. JOSEPH METZGER

Ocean Dynamics and Prediction Branch
Oceanography Division

AND

SUZANNE N. CARROLL

Planning Systems, Inc.
Stennis Space Center, Mississippi

November 5, 2008

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
05-11-2008	Memorandum Report	

4. TITLE AND SUBTITLE

User's Manual for the Polar Ice Prediction System (PIPS) Version 3.0

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
0602435N

6. AUTHOR(S)

Pamela G. Posey, Lucy F. Smedstad, Ruth H. Preller, E. Joseph Metzger, and Suzanne N. Carroll*

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER
73-6057-08-5

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Naval Research Laboratory
Oceanography Division
Stennis Space Center, MS 39529-5004

8. PERFORMING ORGANIZATION REPORT NUMBER

NRL/MR/7320--08-9154

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Office of Naval Research
One Liberty Center
875 North Randolph St.
Arlington, VA 22203-1995

10. SPONSOR / MONITOR'S ACRONYM(S)
ONR

11. SPONSOR / MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

*Planning Systems, Inc., MSAAP Building 9121, Stennis Space Center, MS 39529

14. ABSTRACT

The Polar Ice Prediction System (PIPS) Version 3.0 is a dynamic sea-ice model that forecasts conditions in all sea-ice covered areas in the northern hemisphere (down to 30° north in latitude). It has a horizontal resolution of approximately 9 km. The vertical resolution in the model has been set at 45 levels so that Arctic shelves, continental slopes, and submarine ridges are accurately represented. Currently, the domain includes the Irminger, Labrador, North, and Baltic Seas on the Atlantic side and the Bering Sea, Sea of Japan, and the Sea of Okhotsk on the Pacific. The PIPS 3.0 system is based on the Los Alamos ice model and coupled (via file transfer) to the operational, global Navy Coastal Ocean Model (gNCOM). The system forecasts daily ice thickness, concentration, and drift in the Arctic Ocean. This report documents the setup and execution of the PIPS 3.0 system.

15. SUBJECT TERMS

Ice forecast
Ice drift
Ice edge
Ice model

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Pamela Posey
Unclassified	Unclassified	Unclassified	UL	83	19b. TELEPHONE NUMBER *(include area code)* (228) 688-5596

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39.18

TABLE OF CONTENTS

TABLES AND FIGURES

1.0 INTRODUCTION

The software described in this document is identified as the Polar Ice Prediction System (PIPS) Version 3.0. PIPS 3.0 is a dynamic sea-ice model that forecasts conditions in all sea-ice covered areas in the northern hemisphere (down to 30° north in latitude). It has a horizontal resolution of approximately 9 km. The vertical resolution in the model has been set at 45 levels so that Arctic shelves, continental slopes and submarine ridges are accurately represented. This allows for 17 levels in the upper 300 m of the water column and a maximum layer thickness in the deep ocean of 300 m. The array size is 1280 x 720. Currently the domain includes the Irminger, Labrador, North and Baltic Seas on the Atlantic side and the Bering Sea, Sea of Japan and the Sea of Okhotsk on the Pacific side.

PIPS 3.0 model bathymetry south of 64° N is derived from the following databases: ETOP05 database, Navy Research Laboratory charts and Canadian Hydrographic Service charts. Bathymetry north of 64° N comes from the 2.5 km resolution digital International Bathymetric Chart of the Arctic Ocean (IBCAO).

The PIPS 3.0 system uses the Los Alamos ice model, CICE (version 3.1), containing improved procedures for model thermodynamics, physics parameterizations and energy based ridging [6], [7] and [13]. It has the ability to predict multi-category ice thickness. The CICE model is presently being coupled (via file transfer) to the operational, global Navy Coastal Ocean Model (NCOM), to predict ice thickness, concentration, and drift in the Arctic Ocean. NCOM is a baroclinic, hydrostatic, Boussinesq; free-surface ocean model that allows its vertical coordinate to consist of sigma coordinates for the upper layers and z-levels below a user-specified depth [4], [5] and [12]. NCOM runs operationally at NAVOCEANO at a resolution of 1/8 degree globally. PIPS 3.0 also forecasts surface ocean current and temperature in the surrounding seas. The scripts shown in the main document reflect how the sea ice model is run and not the coupled system. Appendix D describes how files are transferred between the ice model and the NCOM model for the daily coupled forecast run.

PIPS 3.0 is driven by heat fluxes and surface winds from the Navy Operational Global Atmospheric Prediction System (NOGAPS). Daily ice concentration updates are accomplished through an objective analysis of ice concentration data from the Special Sensor Microwave/Imager (SSM/I) located on the Defense Meteorological Satellite Program (DMSP) satellite.

There are four primary components that work together to comprise the PIPS 3.0 model:

- a thermodynamic model that calculates snowfall as well as local growth rates of snow and ice due to vertical conductive, radiative and turbulent fluxes;
- an ice dynamics model, which predicts the velocity field of the ice pack based on a model of the material strength of the ice;
- a transport model that depicts advection of the aerial concentration, ice volumes and other state variables;

Manuscript approved October 3, 2008.

- a ridging parameterization that transports ice among thickness categories based on energetic balances and rates of strain.

1.2 Document Overview

The purpose of this Software User's Manual (SUM) is to describe the setup and execution of the Polar Ice Prediction System (PIPS) Version 3.0. Because the PIPS 3.0 model is largely based on the CICE model, this document reflects the information found in the Los Alamos Sea Ice (CICE) Software User's Manual [1]. This User's Manual includes the installation, setup and execution of the model specific to the needs and environment of the Naval Oceanographic Office. This document, along with a Software Design Description [2] and a Validation Test Report [3], forms a comprehensive documentation package for the PIPS 3.0 model system.

2.0 APPLICATION

2.1 Description of PIPS 3.0 Usage

This manual describes in detail the procedures for running the Navy Polar Ice Prediction System (PIPS) Version 3.0 at the Naval Oceanographic Office (NAVOCEANO). The program is compiled by running the **comp_ice** script, along with a **Macros.AIX** file, on the IBM "Babbage" system at NAVOCEANO. PIPS 3.0 is configured so that the user can manipulate the input parameters file, **ice_in**, and the variables in **cice3.1_lsf** file to specify the environment for a model run. Contact Pamela Posey at the Naval Research Laboratory for assistance and access to setup files specific to the user's needs.

This User's Manual is largely based on the Los Alamos CICE User's Manual [1]. A separate technical manual compliments this document and contains the mathematical formulation, solution procedure, and code of the model as well as flow charts and descriptions of the programs and sub-programs [2].

Generally speaking, subroutine names are given in *italic* and file names are **boldface** in this document. Symbols used in the code are typewritten, while corresponding symbols in this document are in the *math* font which is similar to *italic*.

2.2 Directory Structure

The present code distribution includes makefiles, input files and several scripts The primary directory is **pips/**, and a run directory (**rundir**) is created upon the initial run of the **comp_ice** script.

+-**pips3/** - primary directory
  +-**README_V3.1** - basic information
  +-**bld/** - makefiles
    +-**Macros.<OS>** - macro definitions for the given operating system, used by Makefile.<OS>
    +-**Makefile.<OS>** - primary makefile for a given operating system (<std> works for most systems)
    +-**makedep.c** - perl script that determines module dependencies
  +-**clean_ice** - script that removes files from the compile directory
  +-**comp_ice** - script that sets up the run directory and compiles the code
  +-**doc/** - documentation
    +-**PIPS_SDD.doc**- software design description
    +-**PIPS_UM.doc**- user's manual
    +-**PIPS_VTR.doc**- validation test report
    +-**cicedoc.pdf** - CICE: the Los Alamos Sea Ice Model Doc and User's Manual

+-**PDF/** - PDF documents of several publications related to CICE
+-**ice.log.<OS>** - sample diagnostic output files
+-**source/** - PIPS 3.0 source code.
+-**CICE.F** - main program
+-**ice_albedo.F** - albedo parameterization
+-**ice_atmo.F** - stability-based parameterization for calculation of turbulent ice-atm fluxes
+-**ice_calendar.F** - keeps track of what time it is
+-**ice_constants.F** - physical and numerical constants and parameters
+-**ice_coupling.F** - interface with the flux coupler
+-**ice_diagnostics.F** - miscellaneous diagnostic and debugging routines
+-**ice_domain.F** - MPI subdomain sizes and related parallel processing info
+-**ice_dyn_evp.F** - elastic-viscous-plastic dynamics component
+-**ice_exit.F** - aborts the model, printing an error message
+-**ice_fileunits.F** - unit numbers for I/O
+-**ice_flux.F** -fluxes needed/produced by the model
+-**ice_flux_in.F** - Reads and interpolates forcing data for stand-alone ice model runs
+-**ice_grid.F** - grid and land masks
+-**ice_history.F** - netCDF output routines and restart read/write
+-**ice_init.F** - namelist and initializations
+-**ice_itd.F** - utilities for managing ice thickness distribution
+-**ice_itd_linear.F** - linear remapping for transport in thickness space
+-**ice_kinds_mod.F** - basic defnitions of reals, integers, etc.
+-**ice_mechred.F** - mechanical redistribution component (ridging)
+-**ice_model_size.F** - grid size and number of thickness categories and vertical layers
+-**ice_model_size.F**- specific ice_model_size.F for use by scripts
+-**ice_mpi_internal.F** - utilities for internal MPI parallelization
+-**ice_ocean.F** - mixed layer ocean model
+-**ice_read_write.F** - utilities for reading and writing files
+-**ice_scaling.F** - ice-area scaling of variables for the coupler
+-**ice_state.F** - essential arrays to describe the state of the ice
+-**ice_therm_itd.F** - thermodynamic changes related to ice thickness distribution (post-coupling)
+-**ice_therm_vertical.F** - vertical growth rates and fluxes (pre-coupling thermodynamics)
+-**ice_timers.F** - timing routines
+-**ice_transport_mpdata.F** - horizontal advection via MPDATA or upwind
+-**ice_transport_remap.F** - horizontal advection via incremental remapping
+-**ice_work.F** - globally accessible work arrays
+-**rundir/** - execution or "run" directory generated when the code is compiled

using the comp_ice script
+-**cice** - code executable
+-**compile/** - directory containing object files, etc.
+-**grid** - horizontal grid file from pips/input_templates/
+-**ice.log.[ID]** - diagnostic output file
+-**ice_in** - namelist input data from pips/input_templates
+-**hist/iceh.[timeID].nc** - monthly average output history file
+-**kmt** - land mask file from pips/input_templates/
+-**run_ice** - batch run script file from pips/input_templates/

2.3 Running Environment

The requirements to run a simulation are limited to the model's compile script **comp_ice**, the model run script **cice3.1_lsf** , macro files for compiling on the IBM systems, makefiles, and the model input parameter file, **ice_in**. Therefore, it may be convenient to set up the input files and look at the model output on a computer separate from the one on which the model itself is being run (e.g., where interactive plotting is available to make it easier to inspect the fields).

The input/output files are IEEE binary, ASCII files or netcdf format and should be fully portable to the different computers that are normally used (Sun, SGI, Cray T3E, IBM-SP, DecAlpha).

2.3.1 PIPS 3.0 Compile Script (*comp_ice*)

The **comp_ice** script compiles the PIPS 3.0 code for execution. The variables at the beginning of the script may be changed for tailoring a specific model run. The variables seen in the sample **comp_ice** script below, however, are common to a run performed at NAVOCEANO. This script has been tested on NAVOCEANO's "BABABGE" machine.

```
#!/bin/csh
#
#@ job_name = pips3.1
#@ output = $(job_name).log
#@ error = $(job_name).log
#@ restart = yes
#@ job_type = parallel
#@ network.MPI = csss,not_shared,US
#@ environment = MP_EUILIB=us
#@ node = 4
#@ total_tasks = 32
#@ node_usage = not_shared
#@ resources = ConsumableCpus(1) ConsumableMemory(500mb)
#@ wall_clock_limit = 2:00:00
```

```
#@ account_no = NRLSS015
#@ class = batch
#@ queue
set echo
setenv SITE NAVO
setenv SYSTEM_USERDIR /scr/posey/pips3

# MPI runs
setenv NX 2
setenv NY 16
setenv BINTYPE MPI
# Set SRCDIR and EXEDIR to your own paths!
setenv SRCDIR /scr/posey/pips3
setenv EXEDIR $SYSTEM_USERDIR/rundir.pips.v3.1 ; if !(-d $EXEDIR) mkdir
-p $EXEDIR

setenv CBLD   $SRCDIR/bld
setenv OBJDIR $EXEDIR/compile     ; if !(-d $OBJDIR) mkdir -p $OBJDIR
setenv RSTDIR $EXEDIR/restart     ; if !(-d $RSTDIR) mkdir -p $RSTDIR

setenv ARCH `uname -s`
echo ARCH, $ARCH
if ( $ARCH == 'UNICOS/mp') setenv ARCH UNICOS

cd $EXEDIR

if !(-e grid)             cp $SRCDIR/regional.cice.r grid
if !(-e ice_in)           cp $SRCDIR/ice_in ice_in
if !(-e run_ice)          cp $SRCDIR/run_ice run_ice

cd $RSTDIR

if !(-e ice.restart_file) cp $SRCDIR/ice.restart_file .

cd $OBJDIR

# Filepath: List of source code directories (in order of importance).
echo $SRCDIR
cat >! Filepath << EOF
$SRCDIR/source
EOF

cc -o makdep $CBLD/makdep.c                                || exit 2
gmake VPFILE=Filepath EXEC=$EXEDIR/cice NX=$NX NY=$NY \
-f  $CBLD/Makefile MACFILE=$CBLD/Macros.$ARCH || exit 2
cd $SRCDIR
```

2.3.2 Run_ice Script (*cice3.1_lsf*)

The **cice3.1_lsf file** is the script used to run PIPS 3.0 at NAVOCEANO. Once PIPS 3.0 is run, data files from PIPS 3.0 are then processed and then used by NCOM to make the next ocean run.

```
#!/bin/csh
#
#@ job_name = cice31.72
#@ output = $(job_name).log
#@ error = $(job_name).log
#@ restart = yes
#@ job_type = parallel
#@ network.MPI = csss,not_shared,US
#@ environment = MP_EUILIB=us
#@ node = 4
#@ total_tasks = 32
#@ node_usage = not_shared
#@ resources = ConsumableCpus(1) ConsumableMemory(500mb)
#@ wall_clock_limit =24:00:00
#@ account_no = NRLSS015
#@ class = standard
#@ queue
set echo
limit stacksize 2000000
cd /scr/posey/pips3
set  stamp   = `date -u '+%y%m%d%H%M'`
 setenv ICE_LOG_FILE cice31.${stamp}
/usr/bin/poe cice31 >&! $ICE_LOG_FILE
#  Exit
#
rm core*
exit 0
```

There are several directories and files created once PIPS 3.0 is compiled on a machine. These are defined in Table 1 below.

Directory/File	Description
rundir/	execution or "run" directory
cice*	code executable
grid	horizontal grid file
ice.log.<OS>	sample diagnostic output files
ice_in	namelist input file
iceh.[timeID].incond.nc	output history file for the initial condition
iceh.[timeID].nc	output history file
kmt	land mask file from
run_ice	batch run script from
rundir/compile/	directory containing object files, etc.
rundir/restart/	restart directory
iced	initial condition
ice.restart_file	restart pointer

Table 1: Files and directories created once PIPS 3.0 is compiled.

2.3.3 PIPS 3.0 Macros File

Macros.AIX.txt is a file of macros necessary for compiling the PIPS 3.0 code on the NAVOCEANO IBM platform "Babbage". An example of the entire Macros.AIX file can be found in Appendix A. The primary macros available in this file are summarized in the table below.

Macro	Description
`-lmass`	IBM - tuned intrinsic library
`-qsmp=noauto`	enables SMP directives, but does not add any
`-qstrict`	don't turn divides into multiplies, etc
`-qhot`	higher-order -transformations (eg. loop padding)
`-qalias=noaryoverlp`	assume no array overlap with respect to equivalence, etc
`-qmaxmem=1`	memory available to compiler during optimization
`-qipa=level=2`	InterProcedure Analysis (eg. inlining) => slow

Macro	Description
	compiles
-p –pg	enable profiling (use in both FFLAGS and LDFLAGS)
-qreport	for smp/omp only
-bmaxdata:0x80000000	use maximum allowed data segment size
-g	always leave it on because overhead is minimal
-qflttrap=...	enable default sigtrap (core dump)
-C	runtime array bounds checking (runs slow)
-qinitauto=...	initializes automatic variables

Table 2: PIPS 3.0 macros from Macros.AIX.txt file.

2.3.4 PIPS 3.0 Makefile

A makefile is used in the execution of PIPS 3.0 at NAVOCEANO. The entire makefile is shown in Appendix B.

Command-line variables

1. MACFILE=<file> ~ The macros definition file to use/include in a run.
2. EXEC=<name> ~ The name given to an executable. The default is *a.out*.
3. VPATH=<vpath> ~ VPATH , default is . (*cwd* only).
4. SRCS=<files> ~ A list of source files. The default is all *.c .F .F90* files in VPATH.
5. VPFILE=<file> ~ A file with a list of directories. It is used to create VPATH.
6. SRCFILE=<file> ~ A file with a list of source files. It is used to create SRCS.
7. DEPGEN=<exec> ~ A dependency generator utility, with a default of *makdep*.
8. <macro defns> ~ Any macro definitions found in this file or the included MACFILE will be over-ridden by command line macro definitions.
9. MODEL=<model> ~ A standard macro definition, often found in the included MACFILE. It is used to trigger special compilation flags.

Usage examples:

```
% gmake MACFILE=Macros.AIX VPFILE=Filepath MODEL=ccm3 EXEC=atm
% gmake MACFILE=Macros.AIX VPFILE=Filepath SRCFILE=Srclist EXEC=pop
% gmake MACFILE=Macros.C90 VPATH="dir1 dir2" SRCS="file1.c file2.F90"
% gmake MACFILE=Macros.SUN SRCS="test.F"
```

2.3.5 PIPS 3.0 Input File Namelist Parameters

2.3.5.1 Ice Namelist (ice_nml)

The **ice_in.txt** file is an input file of namelist parameters used in running the PIPS 3.0 model. The variables are described in the order they appear in the **ice_in.txt**.

Name	Type/Options	Description	Default Values / Directory Location
year_init	yyyy	The initial year, if not using restart	= 0001
istep0	integer	Initial time step number	= 0
dt	seconds	Thermo/transport time step length	= 2800.
ndte	integer	Number of EVP subcycles	= 120
npt	integer	Total number of time steps to take	= 70
diagfreq	integer	Frequency of diagnostic output in dt eg., 10 is once every 10 time steps	= 30
histfreq	y, m, w, d, h	Write history output once a year, month, week, day, or every time step	= 'h'
dumpfreq	y, m, d	Write restart every dumpfreq_n years, months, days	= 'd'
dumpfreq_n	integer	Frequency restart data is written	= 1
hist_avg	true/false	Write time-averaged data if true Write snapshots of data if false	= .true.
restart	true/false	Initialize using restart file	= .true.
print_points	true/false	Print diagnostic data for two grid points	= .true.
print_global	true/false	Print diagnostic data, global sums	= .true.
kitd	0 /1	If 0, delta function ITD approx. If 1, linear remapping ITD approx.	= 1
kcatbound	0/1	If 0, original category	= 1

Name	Type/Options	Description	Default Values / Directory Location
		boundary formula If 1, new category boundary formula	
kdyn	0 /1	If 0, EVP dynamics OFF If 1, EVP dynamics ON	= 1
kstrength	0 /1	If 0, [5] ice strength formulation If 1, [13] ice strength formulation	= 1
krdg_partic	0/1	If 0, old ridging participation function If 1, new ridging participation function	= 1
krdg_redist	0/1	If 0, old ridging redistribution function If 1, new ridging redistribution function	= 1
evp_damping	true/false	If true, damp elastic waves, [6]	= .false.
advection	remap, mpdata, upwind	remap: Linear remapping advection mpdata: 2nd order MPDATA upwind: 1st order MPDATA	= 'remap'
grid_type	rectangular, displaced_pole pips	rectangular: Defined in *rectgrid* pips: Read from file in *pipsgrid*	= 'pips'
grid_file	filename	Name of grid file to be read	= 'grid_cice_1280x720.r'
kmt_file	filename	Name of land mask file to be read	= 'kmt'
dump_file	filename prefix	Output file for restart dump	= 'iced'
restart_dir	path/	path to restart directory	= '/scr/posey/pips3c/'
pointer_file	pointer filename	Contains restart filename	= '/scr/posey/pips3c/ice.restart_file'
hist_dir	path/	path to history output directory	= '/scr/posey/pips3c/'
history_file	filename prefix	Output file for history	= 'iceh'

Name	Type/Options	Description	Default Values / Directory Location
diag_file	filename	Diagnostic output file	= 'ice_diag.d'
oceanmixed_ice	true/false	Active ocean mixed layer calculation	= .true.
albicev	$0 < \alpha < 1$	Visible ice albedo for thicker ice	= 0.65
albicei	$0 < \alpha < 1$	Near infrared ice albedo for thicker ice	= 0.65
albsnowv	$0 < \alpha < 1$	Visible, cold snow albedo	= 0.85
albsnowi	$0 < \alpha < 1$	Near infrared, cold snow albedo	= 0.85
ycycle	integer	No. of years in forcing data cycle	= 1
fyear_init	yyyy	First year of atmospheric forcing data	= 2008
atm_data_dir	path/	Path to atm forcing data directory	= '/scr/posey/pips3c/data_in/'
ocn_data_dir	path/	Path to oceanic forcing data directory	= '/scr/posey/pips3c/data_in/'

Table 3: Ice namelist parameters.

2.3.5.2 Ice Fields Namelist (icefields_nml)

Name	Description	Default Values
f_hi	Ice thickness	= .true.
f_hs	Snow thickness	= .true.
f_Tsfc	Temperature of ice/snow top surface (in category *n*)	= .false.
f_aice	Ice concentration	= .true.
f_uvel	*x*-component of velocity	= .true.
f_vvel	*y*-component of velocity	= .true.
f_fswdn	Incoming shortwave radiation down	= .true.
f_flwdn	Incoming longwave radiation down	= .false.
f_snow	Snowfall rate	= .false.

Name	Description	Default Values
f_snow_ai	Snowfall rate weighted by aice	= .false.
f_rain	Rainfall rate	= .false.
f_rain_ai	Rainfall rate weighted by aice	= .false.
f_sst	Sea surface temperature	= .true.
f_sss	Sea surface salinity	= .true.
f_uocn	Ocean current, *x* direction	= .true.
f_vocn	Ocean current, *y* direction	= .true.
f_frzmlt	Freezing/melting potential	= .false.
f_fswabs	Absorbed shortwave radiation	= .false.
f_fswabs_ai	Absorbed shortwave radiation weighted by aice	= .false.
f_albsni	Snow/ice broad band albedo	= .false.
f_alvdr	Visible direct albedo	= .false.
f_alidr		= .false.
f_flat	Latent heat flux	= .false.
f_flat_ai	Latent heat flux weighted by aice	= .false.
f_fsens	Sensible heat flux	= .false.
f_fsens_ai	Sensible heat flux weighted by aice	= .false.
f_flwup	Incoming longwave radiation upward	= .false.
f_flwup_ai	Incoming longwave radiation upward weighted by aice	= .false.
f_evap	Evaporation water flux	= .false.
f_evap_ai	Evaporation water flux weighted by aice	= .false.
f_Tref	2m atmospheric reference temperature	= .false.

Name	Description	Default Values
f_Qref	2 m atmospheric reference specific humidity	= .false.
f_congel	Basal ice growth	= .false.
f_frazil	Frazil ice growth	= .false.
f_snoice	Snow-ice formation	= .false.
f_meltt	Top ice melt	= .false.
f_meltb	Basal ice melt	= .false.
f_meltl	Lateral ice melt	= .false.
f_fresh	Fresh water flux to ocean	= .false.
f_fresh_ai	Fresh water flux to ocean weighted by aice	= .false.
f_fsalt	Net salt flux to ocean	= .false.
f_fsalt_ai	Net salt flux to ocean weighted by aice	= .false.
f_fhnet	Net heat flux to ocean	= .false.
f_fhnet_ai	Net heat flux to ocean weighted by aice	= .true.
f_fswthru	Shortwave penetrating to ocean	= .false.
f_fswthru_ai	Shortwave penetration to ocean weighted by aice	= .false.
f_strairx	Stress on ice by air in the *x*-direction (centered in U cell)	= .true.
f_strairy	Stress on ice by air in *y*-direction (centered in T cell)	= .true.
f_strtltx	Surface stress due to sea surface slope in x-direction	= .false.
f_strtlty	Surface stress due to sea surface slope in y-direction	= .false.
f_strcorx	Coriolis stress (x)	= .true.
f_strcory	Coriolis stress (y)	= .true.
f_strocnx	Ice-ocean stress, *x* dir. (U	= .true.

Name	Description	Default Values
	cell)	
f_strocny	Ice-ocean stress, *y*-dir. (T cell)	= .true.
f_strintx	Divergence of internal ice stress, *x*-direction	= .true.
f_strinty	Divergence of internal ice stress, *y*-direction	= .true.
f_strength	Ice strength (pressure)	= .true.
f_opening	Lead area opening rate	= .false.
f_divu	Strain rate I component, velocity divergence	= .false.
f_shear	Strain rate II component	= .false.
f_sig1	Principal stress components (diagnostic)	= .false.
f_sig2	Principal stress components (diagnostic)	= .fasle.
f_dvidtt	Ice volume tendency due to thermodynamics	= .false.
f_dvidtd	Ice volume tendency due to dynamics/transport	= .false.
f_daidtt	Ice area tendency due to thermodynamics	= .false.
f_daidtd	Ice area tendency due to dynamics/transport	= .false.
f_mlt_onset	Day of year that surface melt begins	= .false.
f_frz_onset	Day of year that freezing begins	= .false.
f_dardg1dt	Ice area ridging rate	= .false.
f_dardg2dt	Ridge area formation rate	= .false.
f_dvirdgdt	Ice volume ridging rate	= .false.
f_hisnap	Ice volume snapshot	= .false.
f_aisnap	Ice area snapshot	= .false.
f_aice1 (through 5)	Ice concentration in grid cell in categories 1 through	= .true.

Name	Description	Default Values
	5	
f_aice6 (through 10)	Ice concentration in grid cell in categories 6 through 10	= .false.
f_vice1 (through 5)	Volume per unit area of ice in categories 1 through 5	= .true.
f_vice6 (through 10)	Volume per unit area of ice in categories 6 through 10	= .false.

Table 4: Ice fields namelist parameters

2.4 Code Modifications

This current model release is PIPS Version 3.0. It replaces PIPS 2.0, which has been operational since 1996. Although the model physics is similar to that of PIPS version 2.0, the ice code has changed considerably. The following is a list of the major changes:

1. Many modules have been rewritten to run efficiently on vector platforms such as the Cray X1 and the Earth Simulator. Shorter loops over categories and vertical layers have been moved outside longer horizontal (*i,j*) loops. Single-column state variables (of dimension *ncat*) have been eliminated. Directives have been put in place to enforce vectorization over certain loops. Nested *i,j* loops have in many cases been joined into a single loop to increase vector length.
2. The mechanical redistribution module, **ice_mechred.F**, has been altered to run more stably in swiftly deformatting regions.
3. The dynamics scheme now handles lower concentration regions more accurately, consistent with the free drift theory.
4. The MPDATA and incremental remapping transport schemes, previously in **ice_transport.F**, have been separated into two routines, **ice transport mpdata.F** and **ice_transport_remap.F**. The remapping module has been revised for efficiency and user-friendliness. For example, it is simpler now to transport additional tracers specified by the user.
5. The open water fraction is now transported horizontally.
6. The two thermodynamics routines have been reorganized. Thermodynamic computations before the *to_coupler* call are now in **ice therm vertical.F**, and computations after the *to_coupler* call are in **ice_therm_itd.F**. These modules replace **ice_therm.F** and **ice_therm_driver.F**.
7. The lateral melting scheme from the CCSM Community Sea Ice Model (CSIM) has been added to PIPS 3.0.
8. Helpful utilities have been added to **ice_itd.F**.
9. Several history fields and namelist options have been added.

10. Additional global diagnostics can be written out. These diagnostics are used for examining conservation of heat, water, and salt.
11. One restart variable (*fhocn*) has been taken out, and another (*fsalt*) has been added. Therefore, **ice_history.F** must be modified if starting a PIPS 3.0 run with a restart file generated by version 2.0.
12. The coupling routine, **ice_coupling.F**, has been updated to conform to the latest CCSM coupler. Two more fields, *Qrefand* and *fswabs*, are now passed to the coupler.
13. The routine that reads forcing data, **ice_flux_in.F**, has been rewritten so that it is more tailored to read data in different formats. Temporal interpolation is now linear instead of cubic.
14. Standard ProTeX prologues have been added to the beginning of each module and subroutine. The prologues permit automated generation of LaTeX documentation from Fortran 90 code.
15. Several minor bugs have been fixed.
16. Many variables and subroutines have been renamed to enhance their description and consistency.
17. Code formatting has been standardized, and some outdated coding practices have been eliminated.

3.0 LIMITATIONS AND ASSUMPTIONS

The user must be aware of the following model limitations before completing a PIPS 3.0 model run.

1. Fluxes sent to the coupler could have incorrect values in grid cells that fluctuate from an ice-free state to having ice during the given time step, or vice versa, due to scaling by the ice area. The flux coupler authors insist on area scaling so that the ice and land models are considered consistently in the coupler (Note that, unlike the ice area, the land area does not suddenly become zero in a grid cell).
2. A sizable fraction (more than 10%) of the total shortwave radiation is absorbed at the surface but preferably should be penetrating into the ice interior. This is due to use of the aggregated, effective albedo instead of the bare ice albedo when `snowpatch` < 1, and solving the problem will require more albedo arrays to be added to the code.
3. The date-of-onset diagnostic variables, *melt_onset* and *frz_onset*, are not included in the restart file. They, therefore, may be incorrect for the current year if the run is restarted after January 1. Also, these variables were applied with the Arctic in mind and may be incorrect for the Antarctic.
4. Timers are architecture dependent.
5. Local domains are not padded for uneven partition of the global domain.

4.0 OPERATING GUIDELINES

PIPS 3.0 is written in fixed-format FORTRAN90 and runs on UNIX host platforms, including SGI Origin 3000 (<**OS> = IRIX64** below), SGI Altix (**Linux**), IBM Power4 (**AIX**) and Cray X1 (**UNICOS**). The code is parallelized through grid decomposition with MPI for message passing between processors, with four processors allocated to each hemisphere. The code has been optimized for vector architectures and tested on Fujitsu VPP 5000, Cray X1, and NEC platforms. At NAVOCEANO, PIPS 3.0 is run using 32 processors on an IBM platform with a ConsumableMemory of 500 mb. With respect to hardware resources, a one-day run of PIPS 3.0 at NAVOCEANO requires 0.85 Processor Hrs.

4.1 Setting Up the Grid, Boundary Conditions and Masks

The spatial discretization is customized for a generalized orthogonal B-grid as in [15] or [10]. The ice and snow area, volume and energy are provided at the center of the cell, velocity is defined at the corners, and the internal ice stress tensor takes four different values within a grid cell. Bilinear approximations are used for the stress tensor and the ice velocity across the cell, as given in [9]. This tends to sidestep the grid decoupling problems associated with the B-grid.

Since ice thickness and thermodynamic variables such as temperature are given in the center of each cell, the grid cells are referred to as "T cells." Occasionally we make reference to "U cells," which are centered on the northeast corner of the corresponding T cells, and each have velocity in their centers. The velocity components are aligned along grid lines.

Generally, the global gridded domain is `imt_globalxjmt_global`, while the subdomains found in the MPI grid decomposition are `imt_localxjmt_local`. The physical part of a subdomain is indexed as [`ilo:ihi,jlo:jhi`], with `num_ghost_cells` "ghost" cells lying outside the domain being used for boundary conditions. These parameters are shown in one dimension in Figure 1. The routines *global_scatter* and *global_gather* allocate information from the global domain to the local domains and back again, respectively. If MPI is not being used for grid decomposition in PIPS 3.0, these routines basically switch the indexing on the global domain to the single, local domain index coordinates. It is recommended that the user select the number of local domains so that the global domain is evenly divided. If the global domain is not evenly distributed among the number of processors, then the last subdomain will contain nonphysical points, or "padding". In addition to a loss of efficiency due to computing at these points, other issues may arise due to incompatible initializations and spurious data values.

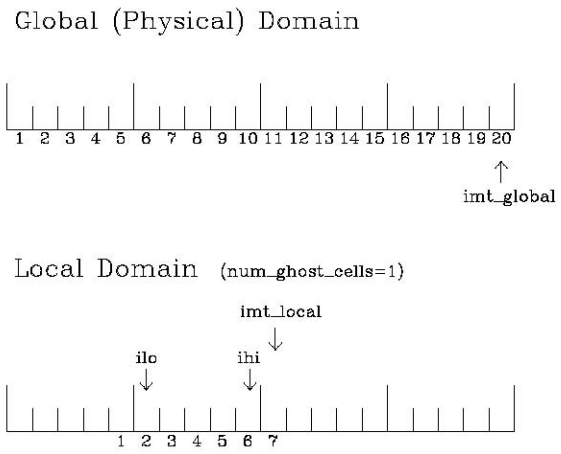

Figure 1: Grid parameters for a sample one-dimensional, 20-cell global domain decomposed into four local subdomains. Each local domain includes one ghost cell on each side, and the physical portion of the local domains are denoted `ilo:ihi`. The parameter `imt_local` is defined as the total number of cells in the local domain, including ghost cells. The same numbering system is applied to each of the four subdomains.

The user may choose from three grid routines: *pipsgrid* reads grid lengths and other parameters for a nonuniform, *popgrid* or *tripole* grid. The input file **grid_cice_1280x720** holds PIPS grid and land mask information. This is a binary unformatted, direct access files generated on an SGI (Big Endian).

In the present implementation with a bipolar, displaced-pole grid, no less than one row of grid cells along the north and south boundaries are understood to be located on land. Along domain boundaries not masked by land, periodic conditions envelop the domain around the globe. The original boundary routine is *bound*. The other boundary routines enhance parallel performance by not filling all four boundaries when that is unwarranted, and by updating multiple spatial arrays at one time. The boundary routines also perform boundary communications between local domains when MPI is in use.

A land mask `hm` (M_h) is specified at the cell centers, with 0 for land and 1 for ocean cells. A corresponding mask `uvm` (M_u) for velocity and other corner quantities is given by

$$M_u(i, j) = \min\{M_h(l), l = (i, j), (i+1, j), (i, j+1), (i+1, j+1)\}.$$

The logical masks `tmask` and `umask` (which are consistent with the real masks `hm` and `uvm`, respectively) are of use in conditional statements.

In addition to the land masks, two other masks are employed in *evp_prep* in order to reduce the dynamics component's role on a global grid. At each time step the logical masks `ice_tmask` and `ice_umask` are established from the current ice extent, so that they have the value ``true'' wherever ice exists. These masks also include a border of cells

surrounding the ice pack for numerical purposes. The logical masks are used in the dynamics component to prevent unnecessary computations on grid points where no ice exists. They are not used in the thermodynamics component, so that ice may form in formerly ice-free cells. Like the land masks `hm` and `uvm`, the ice extent masks `ice_tmask` and `ice_umask` are for T cells and U cells, respectively.

Two extra masks are created for the user's convenience: `mask_n` and `mask_s` may be used to compute or write data specific to the northern or southern hemispheres, respectively.

4.2 Initialization and Coupling

The PIPS 3.0 parameters and variables are initialized in many stages. Several constants and physical parameters are set in **ice_constants.F**. Namelist variables (see Appendix B) are handled in the subroutine *input_data* (see Appendix B) and values may be changed at run time. The namelist variables are assigned default values in the code, which may then be altered once the input file **ice_in** is read. Many of the variables provided in the namelist declaration given in **ice_init.F** are not usefully implemented in the current version of PIPS 3.0, but these variables are used in the NCAR CCSM ice model and are mentioned in the namelist declaration for consistency with that code. Physical constants, variables, and numerical parameters are first set in initialization routines for each ice model component or module. Then, if PIPS 3.0 is being restarted from a previous run, some variables are then read and reinitialized in the *restartfile* subroutine. Lastly, albedo is initialized based on the initial ice state. Some of these parameters will be explained in more detail in Appendix B.

The ice component corresponds with the flux coupler by passing messages using MPI, which is started in the *setup_mpi* subroutine for both coupled and stand-alone MPI runs. Further initialization for coupling takes place in *ice_coupling_setup* and *init_cpl*. The subroutines *to_coupler* and *from_coupler* respectively pack and unpack the data being passed between the ice component and the flux coupler, and perform any needed averaging and unit conversions.

For stand-alone runs, routines in the **ice_flux_in.F** module read and interpolate data from files. They are intended solely to provide guidance for the user to write his or her own routines. Whether the PIPS 3.0 code is to be run in stand-alone or coupled mode is decided at compile time, as described below.

4.3 Choosing an Appropriate Time Step

The time step is selected based on the stability of the transport component (both horizontal and in thickness space) and on the resolution of the physical forcing. PIPS 3.0 permits the dynamics, ridging, and advection portion of the code to be run with a shorter timestep, Δt_{dyn} (`dt_dyn`), than the thermodynamics timestep Δt (`dt`). In this case, `dt` and the integer ndyn_`dt` are specified, and dt_`dyn = dt/ndyn_dt`.

A conservative estimate of the horizontal transport time step bound, or CFL condition, under remapping produces

$$\Delta t_{dyn} < \frac{\min(\Delta x, \Delta y)}{2\max(u, v)}.$$

As discussed in Section 5.2.2.3 of the PIPS 3.0 SDD [2] and [11], the maximum time step in practice is typically determined by the time scale for large modifications in the ice strength (which partially depends on wind strength). Using the strength parameterization of [14] limits the time step to 30 minutes for the old ridging scheme, and to two hours for the new ridging scheme, assuming Δx = 10 km. Practical limits could be somewhat less, contingent on the strength of the atmospheric winds.

Transport in thickness space requires a similar restriction on the time step. This is given by the ice growth/melt rate and the smallest scale of thickness among the categories, $\Delta t < \min \Delta H / 2 \max f$, where ΔH is the distance between category boundaries and f represents the thermodynamic growth rate. For the five-category ice thickness distribution employed as the default in this distribution, this is not a strict limitation: $\Delta t < 19.4$ hr, assuming $\max f = 40$ cm/day.

The dynamics component is subcycled `ndte` (N) times per time step so that the elastic waves basically disappear before the next time step. The subcycling time step (Δt_e) is then

```
dte = dt_dyn/ndte.
```

A second parameter, $E_\circ$ (`eyc`), must be chosen, which defines the elastic wave damping timescale T, discussed in Section 5.2.2.4 of the PIPS 3.0 SDD [2] as `eyc*dt_dyn`. The forcing terms are not updated during the subcycling. Given the small step (`dte`) at which the EVP dynamics model is subcycled, E the elastic parameter, is also limited by stability constraints, as discussed in [8]. Linear stability analysis for the dynamics component shows that the numerical method is stable providing the subcycling time step Δt_e sufficiently resolves the damping timescale *T*. For the stability analysis several simplifications of the problem had to be made. Therefore the location of the boundary between stable and unstable regions is simply an estimate. In practice, the ratio $\Delta t_e : T : \Delta t$ = 1 : 40 : 120 supplies both stability and acceptable efficiency for time steps (Δt) on the order of one hour.

Notice that only T and Δt_e figure into the stability of the dynamics component; Δt does not. The thermodynamics module is stable for any time step. Although the time step may not be closely limited by stability considerations, large time steps (*eg.,* $\Delta t = 1$ day, given daily forcing) do not generate accurate results in the dynamics component. The reasons for this error are discussed in [8]; see [10] for its practical effects.

4.4 Model Output

4.4.1 Output Discussion

Model output data is averaged over the period given by `histfreq` and written to netCDF files prepended by `history_file` in **ice_in**. That is to say, if `history_file='iceh'` then the filenames will have the form **iceh.[timeID].nc**. If `history_file='iced'` then restart files are written at the "dump" frequency in **ice_in**. Once the restart files are written, the filename is then written into the file *ice_restart_file*. Header information for data enclosed in these files is displayed with the command `ncdump -h filename.nc`. With this model version, standard ice data fields are output. The user can add (or subtract) variables not readily available in the namelist by following the instructions in **ice_history.F**.

A small number of thermodynamic variables have special `_hist` forms in addition to the standard quantity used in the code. These are variables that are initialized in the middle of the time step (at the start of the second thermodynamics routine, *thermo_itd*), immediately after being sent to the coupler, although they can change at the beginnings of the time step (in *thermo_vertical*). The "standard" variable initialized as such holds a full time step's worth of data when it is sent to the coupler; its history complement is initialized at the beginning of the time step and therefore also contains a full time step's worth of data, although its value might be somewhat different from that sent to the coupler. This code modification was prepared for coupled model load balancing.

The normalized principal components of internal ice stress are calculated in *principal_stress* and written to the history file. This computation is unnecessary for the simulation. Principal stresses are computed merely for diagnostic purposes and included here for the user's convenience.

Like `histfreq`, the parameter `diagfreq` may be used to control how often output is written. In the present PIPS 3.0 release, `diagfreq` is used to determine the frequency that diagnostic data are written to the log file. The log file unit to which diagnostic output is written is established in **ice_fileunits.F**. If `diag_type = 'stdout'`, it is written to standard out (or to **ice.log.[ID]** if standard out is redirected as in **run_ice**). Otherwise it is written to the file provided by `diag_file`. Other than the standard diagnostic output (maximum area-averaged thickness, average albedo, velocity, total ice area, and total ice and snow volumes), the namelist options `print_points` and `print_global` cause extra diagnostic information to be computed and written. The option `print_global` produces global sums that are helpful for checking global conservation of mass and energy. `print_points` writes data for two specified grid points. For the current version of PIPS 3.0, one point is near the North Pole and the other is in the Weddell Sea. These can be changed in **ice_diagnostics.F**.

A binary unformatted file is generated that holds all of the data that PIPS 3.0 uses for a full restart. The filename begins with the character string `dumpfile`, and the restart dump

frequency is provided by `dumpfreq` and `dumpfreq_n`. The pointer to the filename from which the restart data is to be read is established in `pointer_file`.

Timing routines are built into **ice_timers.F**. To use the timers, initialize them first with *ice_timer_clear*, then wrap the portion of code to be timed with *ice_timer_start* and *ice_timer_stop*. Lastly, use *ice_timer_print* to write the results to the log file. Each of these subroutines requires a single argument, the timer number. Calling *ice_timer_clear* or *ice_timer_print* with an argument of -1 starts all of the timers at the same time, or prints all of the timings, instead of calling each individually. Currently, the timers are set up as in Table 5.

Timer Number	Label	Description
0	Total	the entire run
1	TimeLoop	total minus initialization and exit
2	Dynamics	EVP
3	Advectn	horizontal transport
4	Column	all vertical (column) processes
5	Thermo	vertical thermodynamics
6	Ridging	mechanical redistribution
7	Cat Conv	transport in thickness space
8	Coupling	sending/receiving coupler messages
9	ReadWrit	reading/writing files
10	Bound	boundary conditions and subdomain communications

Table 5: PIPS 3.0 timers.

The timings given by these timers are not mutually exclusive. For example, the column timer (4) includes timings from 5, 6 and 7. Subroutine *bound* (timer 10) is called from several places in the code, including the dynamics and advection modules. The timers use *MPI_WTIME* for parallel runs and the F90 intrinsic *system_clock* for single-processor runs.

4.4.2 Types of model output available

1. **Daily Ocean** files which include surface heat flux, surface freshwater flux, sea surface height (SSH; cm), 3D total velocity, u and v (cm/sec), 3D temperature and salinity (T and S) fields (°C and ppt).

2. **Daily Ice** files, such as ice u and v velocity (m/s), ice thickness (m), ice concentration fraction (x 100 to get %), heat flux from ice to ocean (W/m^2), atmosphere/ice stress (N/m^2), and ocean to ice stress (N/m^2).

The daily ice files are ~120 MB following conversion to netCDF format.

4.4.3 Examples of Model Output

The figures shown below provide an example of the PIPS 3.0 model grid domain, as well as just a few of the many examples of graphical output available from the model. Graphics are available in .jpg, .ps and .tif formats.

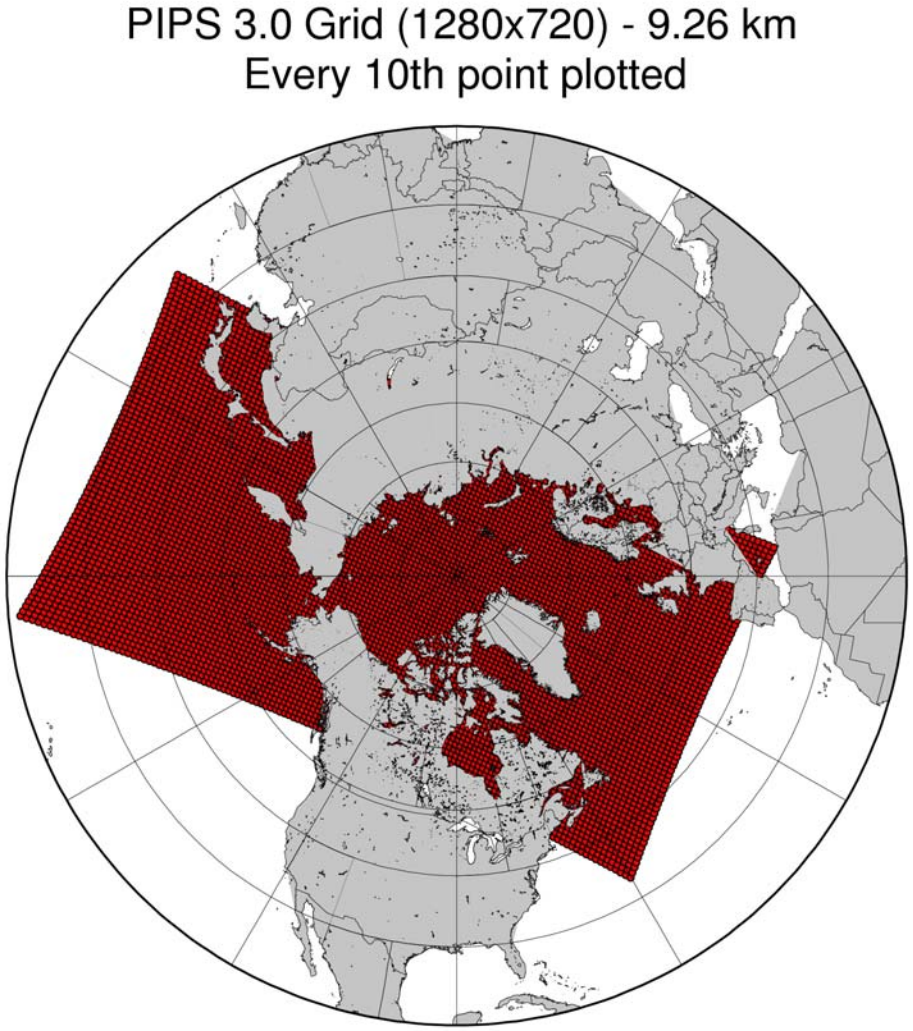

Figure 2: PIPS 3.0 model grid. Unclassified data. Distribution unlimited.

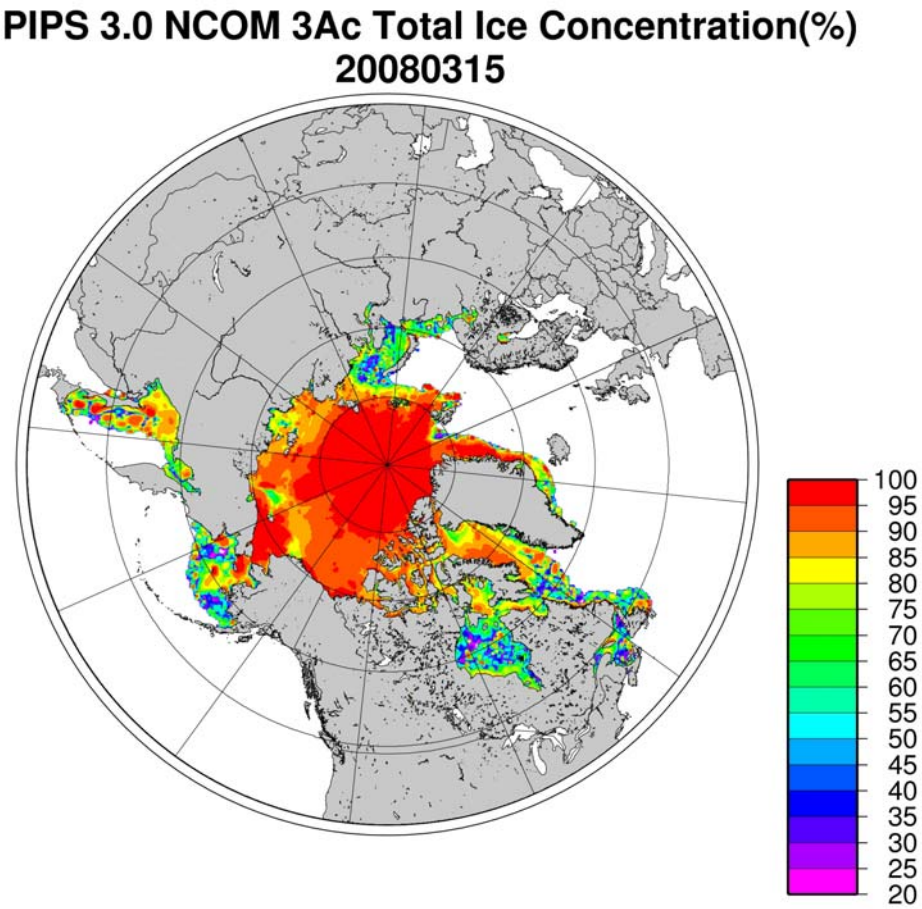

Figure 3: Example daily output of total ice concentration (%) with PIPS 3.0 coupled with NCOM for March 15, 2008 at the North Pole. Unclassified data. Distribution unlimited.

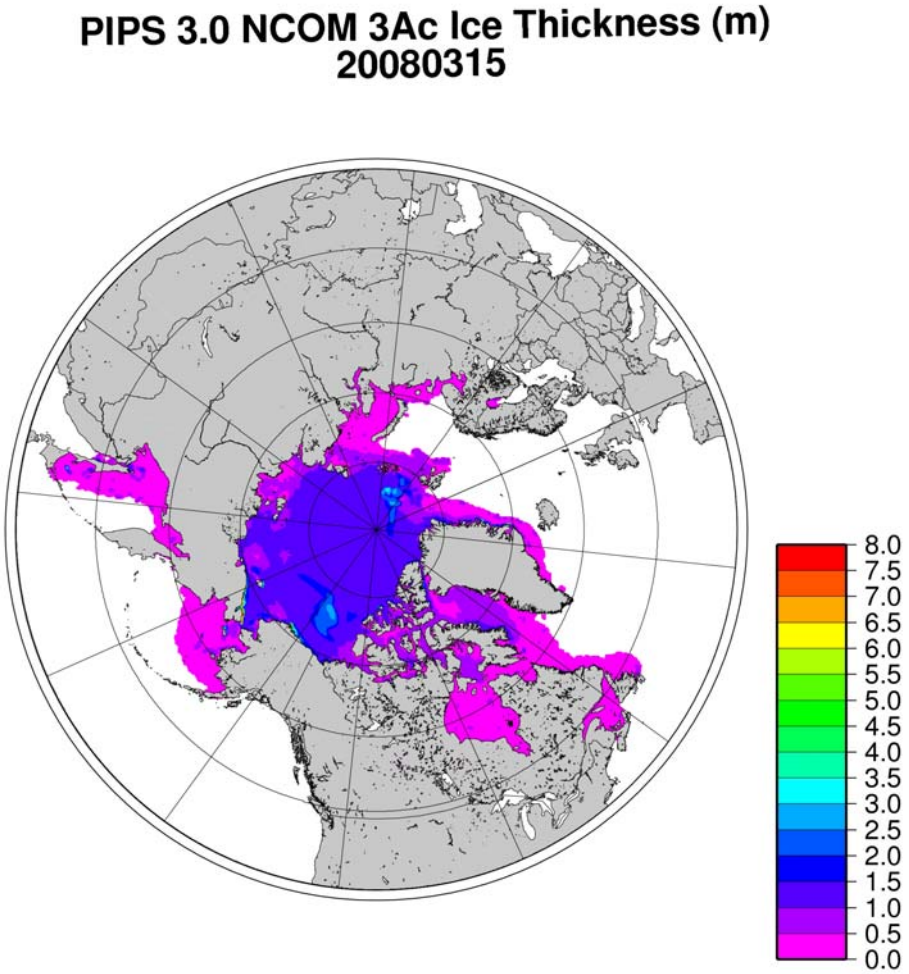

Figure 4: Daily ice thickness in meters for PIPS 3.0 coupled with NCOM at the North Pole on March 15, 2008. Unclassified data. Distribution unlimited.

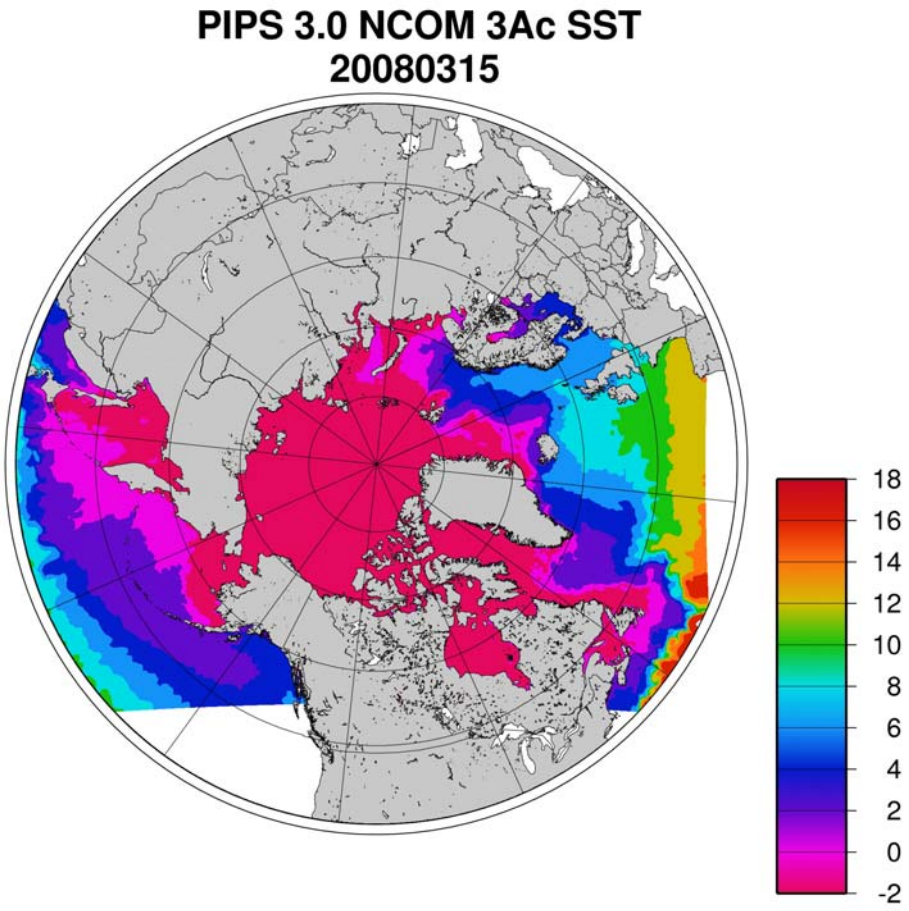

Figure 5: Daily average sea surface temperature (C) for PIPS 3.0 coupled with NCOM on March 15, 2008. Unclassified data. Distribution unlimited.

4.5 Execution Procedures

To compile and execute the code, go to the source directory and

1. Alter directories in the script **comp_ice (see Section 2.3.1)**,
2. Run **comp_ice.** This will set up the run directory and create the executable ' **cice31**,'
3. Alter the script **clean_ice** (see **Appendix C4**) and execute it. This will clean the compile directory and allow the model to start fresh.

In the run directory,

1. Alter `atm_data_dir` and `ocn_data_dir` in the namelist file **ice_in (**see **Appendix C3)**,
2. Alter the script **run_ice** for your system (see **Section 2.3.2**),
3. Execute **run_ice**.

If this fails, see Section 5.1 for troubleshooting the initial setup.

This procedure creates the output log file **ice.log.[ID]**. If `npt` is long enough compared with `dumpfreq` and `histfreq`, then dump files **iced.[timeID]** and netCDF history output files **iceh.[timeID].nc** are created. The log file must be similar to **ice.log.〈OS〉**, which is provided for the user's convenience. These log files were created using MPI on eight processors (`NX=4` and `NY=2`), on the PIPS 3.0 grid.

Several precompiler options are available in **comp_ice** for configuring the run, shown in the table below:

Location	Variable	Options	Description
comp_ice			
	BINTYPE	MPI	use MPI for internal parallelization
	NX, NY	(integers)	number of MPI processors assigned to each coordinate direction

Table 6: Precompiler options in comp_ice for run configuration.

The scripts describe a number of environment variables, typically as directories that are needed to edit your own environment. Two of these environment variables are defined externally, `$HOME`, which points to your home directory (where the PIPS 3.0 directory should be installed), and `$SYSTEM_USERDIR`, which points to scratch disks on the machines at the Naval Oceanographic Office.

PIPS 3.0 namelist variables available for changes after compile time are seen in **ice.log.***

with values read in from the file **ice_in**; Definitions are given in Appendix B. For instance, to run for a different length of time, say two days, set `npt=60` in **ice_in**. Presently, the user furnishes the time step `dt`, the number of dynamics/advection/ridging subcycles `ndyn_dt`, and the number of evp subcycles `ndte`, and dte is then calculated in subroutine *init_evp*. The main reason for doing it this way is to guarantee that `ndte` is an integer.

To restart from a previous run, set the filename in **ice.restart_file** (created by the previous run) to the desired data file (**iced.[timeID]**), then set `restart=.true.` in **ice_in**. Restarts are exact for MPI or single processor runs.

The structure and flow of the PIPS 3.0 code are quite well outlined in the main driver routine **CICE.F**. Notice that the thermodynamics routine is broken into two sections, so that fluxes may be returned to the coupler as soon as possible. This allows the flux coupler to deliver the thermodynamic fluxes to other component models while the ice model keeps running.

5.0 TROUBLESHOOTING

5.1 Initial Setup Issues

The script **comp_ice** (see Section 2.3.1) is configured so that the files **grid**.**kmt**, **ice_in**, **run_ice**, **iced_gx3_v3.1** and **ice.restart_file** are NOT overwritten following the first setup. To make changes to the original files in **input_templates/** rather than to those in the run directory, either remove the files from the run directory before executing **comp_ice** or edit the script.

If the code fails to compile or run, or if the model configuration has changed, try the following:

- Create **Macros.***. **Makefile.*** and **run_ice.*** files for your specific platform, if they do not already exist. Type 'uname -s' at the prompt and compare the result with the file suffixes; (`UNICOS/mp` has been renamed as `UNICOS` for simplicity). Samples of the Macros and Makefile are found in Appendix C. The **run_ice** script is found in Section 2.3.2.
- Modify the `INCLUDE` directory path and other settings for your system in the scripts, as well as in the **Macros.*** and **Makefile.*** files.
- Change directory paths, file names and the execution command as necessary in the **run_ice** and **ice_in** scripts.
- Set the internal parallelization method and number of processors (`BINTYPE`, `NX`, `NY`) in **Macros.***. `NX` and `NY` should evenly divide the respective number of grid points in each direction. Try `NY` $\leq$ 2 for load balancing; otherwise processors assigned subdomains near the equator have little work to do.
- For stand-alone runs, make sure that `-Dcoupled` is *not* set in the **Macros.*** file. For coupled runs, check that `-Dcoupled` and `-DCCSMcoupled` are set in the **Macros.*** file. The option `-Dfcd_coupled` controls another model

configuration not available in this PIPS 3.0 distribution.

• Change the grid size and other parameters in **source/ice_model_size.F**.

5.2 Slow Execution

On some architecture, underflows (10^{-300} for example) are not flushed automatically to zero. Typically a compiler flag is available to do this, but if it is not, then uncomment the block of code at the end of subroutine *stress,* found in **ice_dyn_evp.F**. The extra computations will slow down the run, but it will not be worse than running with the underflows.

5.3 Debugging Hints

Several utilities are available that can be useful when debugging the code. Not all of these will work ubiquitously in the code, due to potential conflicts in module dependencies.

- *debug_ice* (**CICE.F**)- A wrapper for *print_state* that is effortlessly called from numerous points during the timestepping loop (see **CICE.F_debug**).
- *print_state* (ice_diagnostics.F)- Prints the ice state and forcing fields for a given grid cell.
- *ice_global_real_minmax* (ice_mpi_internal.F)- Computes and prints the minimum and maximum values for a real array. A sister routine, *ice_global_real_sum,* prints the global sum of all elements in a real array.
- `diag = .true.` (in calls to *ice_read*)- Prints global max and min values for the field being read.
- `Debug = .true.` (**ice_in**)- Prints many diagnostic quantities for forcing data read in **ice_flux_in.F**.
- `print_global` (**ice_in**) - If true, it computes and prints several global sums for energy and mass balance analysis. This option could significantly degrade code efficiency.
- `print_points` (**ice_in**) - If true, several diagnostic quantities are printed for two grid cells, one close to the North Pole and another in the Weddell Sea. This utility also supplies the local grid indices and processor number (`ip, jp, my_task`) for these points, which can be used in combination with `check_step`, to call *print_state*. These flags are set in **ice_diagnostics.F**. This option can be quite slow, due to gathering data from MPI subdomains.

5.4 Known Bugs

1. Fluxes sent to the coupler could have incorrect values in grid cells that transform from an ice-free state to having ice during the given time step, or vice versa, due to scaling by the ice area. The flux coupler must have area scaling so that the ice and land models are treated reliably in the coupler (but note that the land area does not suddenly become zero in a grid cell, as does the ice area).

2. A significant fraction (more than 10%) of the total shortwave radiation is absorbed at the surface. It should, however, be penetrating into the ice interior instead. This is due to use of the aggregated, effective albedo rather than the bare ice albedo when `snowpatch` < 1. Repairing the problem will require more albedo arrays to be added to the code.
3. The date-of-onset diagnostic variables, `melt_onset` and `frz_onset`, are not included in the restart file. These could therefore be incorrect for the current year if the run is restarted after Jan 1. Also, these variables were employed with the Arctic in mind and may be incorrect for the Antarctic.
4. The single-processor *system_clock* time could give erratic displays on some architecture.
5. Local domains are not padded for uneven partition of the global domain.
6. History files that hold time averaged data (`hist_avg = .true.` in **ice_in**) will be incorrect if the user is restarting from midway through an averaging period.
7. In stand-alone runs, restarts from the end of `ycycle` will not be accurate.

6.0 FUNCTIONAL DESCRIPTION

For a discussion of the functional description see the accompanying Software Design Description (SDD) manual [2].

7.0 TECHNICAL REFERENCES

7.1 PIPS 3.0 Software Documentation

[1] E.C. Hunke and W.H. Lipscomb, "CICE: the Los Alamos Sea Ice Model Documentation and Software", available at http://climate.lanl.gov/Models/CICE/

[2] P.G. Posey, L.F. Smedstad, R.H. Preller, E.J. Metzger and S.N. Carroll. "Software Design Description For The Polar Ice Prediction System (PIPS) Version 3.0", NRL/MR/7320—08-9150, Ocean Modeling Division, Naval Research Laboratory, Stennis Space Center, MS, 2008.

[3] P.G. Posey, L.F. Smedstad, R.H. Preller, E.J. Metzger and S.N. Carroll. "Validation Test Report For The Polar Ice Prediction System (PIPS) Version 3.0", PSI Technical Report SSC-003-06, Ocean Modeling Division, Naval Research Laboratory, Stennis Space Center, MS, 2008.

7.2 General Technical Documentation

[4] C.N. Barron, A.B. Kara, P.J. Martin, R.C. Rhodes, and L.F. Smedstad. Formulation, implementation and examination of vertical coordinate choices in the Global Navy Coastal Ocean Model (NCOM). *Ocean Modelling*, 11:347-375, 2006.

[5] C.N. Barron, A.B. Kara, R.C. Rhodes, C. Rowley and L.F. Smedstad. Validation Test Report for the 1/8° Global Navy Coastal Ocean Model Nowcast/Forecast System. NRL/MR/7320—07-9019, Naval Research Laboratory, Stennis Space Center, MS, 2007.

[6] W.D. Hibler. A dynamic thermodynamic sea ice model. *J*. Phys. Oceanogr., 9:817-846, 1979.

[7] E.C. Hunke. Viscous-plastic sea ice dynamics with the EVP model: Linearization issues. *J. Comput. Phys*. 170:18-38, 2001.

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

[9] E. C. Hunke and J. K. Dukowicz. The Elastic-Viscous-Plastic sea ice dynamics modelin general orthogonal curvilinear coordinates on a sphere—Effect of metric terms. *Mon. Wea. Rev.*, 130:1848– 1865, 2002.

[10] E.C. Hunke and Y. Zhang. A comparison of sea ice dynamics models at high resolution. *Mon. Wea. Rev.*, 127:396–408, 1999.

[11] W.H. Lipscomb, E.C. Hunke, W. Maslowski, and J. Jakacki. Improving ridging schemes for high resolution sea ice models. *J. Geophys. Res.-Oceans*, 112, 2007.

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

[13] R.C. Rhodes, H.E. Hurlburt, A.J. Wallcraft, C.N. Barron, P.J. Martin, E.J. Metzger, J.F. Shriver, D.S. Ko, O.M. Smedstad, S.L. Cross and A.B. Kara. Navy Real-Time Global Modeling Systems. *Oceanography*, 15(1): 29-44, 2002.

[14] D.A. Rothrock. The energetics of the plastic deformation of pack ice by ridging. *J. Geophys. Res.*, 80:4514–4519, 1975.

[15] R.D. Smith, S. Kortas, and B. Meltz. Curvilinear coordinates for global ocean models. Technical Report LA-UR-95-1146, Los Alamos National Laboratory, 1995.

8.0 NOTES

8.1 Acronyms and Abbreviations

Acronym	Definition
CCSM	Community Climate System Model
CICE	Los Alamos Sea-Ice Model
DMSP	Defense Meteorological Satellite Program
DTG	Date-Time-Group
EVP	Elastic-Viscous-Plastic
FNMOC	Fleet Numerical Meteorology and Oceanography Center
HYCOM	HYbrid Coordinate Ocean Model
IBCAO	International Bathymetric Chart of the Arctic Ocean
I/O	Input/Output
ITD	Ice Thickness Distribution model
LANL	Los Alamos National Laboratory
MPDATA	Multidimensional Positive Definite Advection Transport Algorithm
MPI	Message Passing Interface
NAVOCEANO	Naval Oceanographic Office
NCAR	National Center for Atmospheric Research
NCOM	Navy Coastal Ocean Model
NOGAPS	Navy Operational Global Atmospheric Prediction System
NRL	Navy Research Laboratory
PIPS	Polar Ice Prediction System
POP	Parallel Ocean Program model
PSI	Planning Systems, Incorporated

S	Salinity
SDD	Software Design Description
SGI	Silicon Graphics Incorporated
SHEBA	Surface Heat Budget of the Arctic Ocean
SSC	Stennis Space Center
SSM/I	Special Sensor Microwave/Imager
SSS	Sea Surface Salinity
SST	Sea Surface Temperature
SUM	Software Users Manual
T	Temperature

APPENDIX A

PIPS 3.0 Primary Variables and Parameters

The following table defines many of the symbols frequently used in the PIPS 3.0 code. Values appearing in this list are either fixed or recommended; most namelist parameters are indicated (*) with their default values. For other namelist options, see Appendix B. All quantities in the code are expressed in MKS units (temperatures may take either Celsius or Kelvin units).

Name	Description	Default Values
A		
advection	type of advection algorithm used	'remap'
ahmax	thickness above which ice albedo is constant	0.5 m
aice0	fractional open water area	
aice(n)	total concentration of ice in grid cell (in category *n*)	
aice_init	concentration of ice at beginning of *dt* (for diagnostics)	
Ain_min	minimum fractional ice area allowed in each category	
albicei	*near infrared ice albedo for thicker ice	0.36
albicev	*visible ice albedo for thicker ice	0.78
albsnowi	*near infrared, cold snow albedo	0.70
albsnowv	*visible, cold snow albedo	0.98
albocn	ocean albedo	0.06
alpha	floe shape constant for lateral melt	0.66
astar	e-folding scale for participation function	0.05
awtidf	weighting factor for near-ir, diffuse albedo	0.16
awtidr	weighting factor for near-ir, direct albedo	0.31
awtvdf	weighting factor for visible, diffuse albedo	0.24
awtvdr	weighing factor for visible, direct albedo	0.29
ANGLE	for conversions between the ocean grid and lat-lon grids	
ANGLET	ANGLE converted to T-cells	
atm_data_dir	*directory for atmospheric forcing data	

Name	Description	Default Values
avgsiz	number of fields that may be written to history file	91
C		
Cf	ratio of ridging work to PE change in ridging	17.
char_len	length of character variable strings	80
char_len_long	length of longer character variable strings	128
check_step	time step for writing debugging data	
cldf	cloud fraction	
congel	basal ice growth	m
cosw	cosine of the turning angle in water	1.
Cp_air	specific heat of air	1005.0 J/kg/K
Cp_wv	specific heat of water vapor	1.81 x 10^3 J/kg/K
Cp_ice	specific heat of fresh ice	2106. J/kg/K
Cp_ocn	specific heat of sea water	4218. J/kg/K
Cm_to_m	cm to meters conversion	0.01
c<*n*>	real(*n*)	
Cs	fraction of shear energy contributing to ridging	0.5
Cstar	constant in Hibler ice strength formula	20
D		
daidtd	ice area tendency due to dynamics/transport	1/s
daidtt	ice area tendency due to thermodynamics	1/s
dalb_mlt	[see **ice_albedo.F**]	-0.075
dalb_mlti	[see **ice_albedo.F**]	-0.100
dalb_mltv	[see **ice_albedo.F**]	-0.150
dardg1dt	rate of fractional area loss by ridging ice	1/s
dardg2dt	rate of fractional area gain by new ridges	1/s
dvirdgdt	ice volume ridging rate	m/s
dbl_kind	definition of double precision	selected_real_kind (13)
dbug	*write forcing data diagnostics	.false.
Delta	function of strain rates (see Section 5.2.2.4)	
depressT	ratio of freezing temps to salinity of brine	0.054 deg/psu
diag_file	*diagnostic output file (alternative to stdout)	

Name	Description	Default Values
diag_type	*where diagnostic output is written	stdout
diagfreq	*how often diagnostic output is written (10 = once/10 *dt*	24
divu	strain rate I component, velocity divergence	1/s
divu_adv	divergence associated with advection	1/s
Dt	*thermo/transport time step	3600.s
dt_dyn	dynamics/transport time step (Δt_{dyn})	
Dte	subcycling time step for elastic dynamics (Δt_e)	s
dtei	1/*dte*, where *dte* is the EVP subcycling time step	1/s
dT_mlt	[**see ice_albedo.F**]	1. deg
dump_file	* output file for restart dump	
dumpfreq	* dump frequency for restarts, y, m, or d	y
dumpfreq_n	*restart output frequency	1
dragw	drag coefficient for water on ice*ρ_w	0.00536*rhow kg/m^3
dxt	width of T cell (Δx) through the middle	m
dxu	width of U cell (Δx) through the middle	m
Dyt	height of T cell (Δy) through the middle	m
dyu	height of U cell (Δy) through the middle	m
dvidtd	ice volume tendency due to dynamics/transport	m/s
dvidtt	ice volume tendency due to thermodynamics	m/s
E		
ecc	yield curve major/minor axis ratio, squared	4.
eice(n)	energy of melting of ice per unit area (in category *n*)	J/m^2
emissivity	emissivity of snow and ice	0.95
eps04	a small number	10^{-4}
eps11	a small number	10^{-11}
eps12	a small number	10^{-12}
eps13	a small number	10^{-13}
eps15	a small number	10^{-15}
esno(n)	energy of melting of snow per unit area (in category *n*)	J/m^2
evap	evaporative water flux	$kg/m^2/s$

Name	Description	Default Values
evp_damping	*if true, use evp damping procedure [6]	F
eyc	coefficient for calculating the parameter E, 0<*eyc*<1	0.36
F		
fcor	Coriolis parameter	1/s
ferrmax	max allowed energy flux error (thermodynamics)	$1x10^{-3}$ W/m^2
fhnet	net heat flux	W/m^2
fhnet_hist	net heat flux to ocean (*fhnet*) for history	W/m^2
flat	latent heat flux	W/m^2
floediam	effective flue diameter for lateral melt	300. m
Flw	incoming longwave radiation	W/m^2
flwout	outgoing longwave radiation	W/m^2
frain	rainfall rate	$kg/m^2/s$
frazil	frazil ice growth	m
fresh	fresh water flux to ocean	$kg/m^2/s$
fresh_hist	fresh water flux (*fresh*) for history	$kg/m^2/s$
frzmlt	freezing/melting potential	W/m^2
Frz_onset	day of year that freezing begins	
fsalt	net salt flux to ocean	$kg/m^2/s$
fsalt_hist	salt flux to ocean (*fsalt*) for history	$kg/m^2/s$
fsens	sensible heat flux	W/m^2
fsnow	snowfall rate	$kg/m^2/s$
fsnowrdg	snow fraction that survives in ridging	0.5
Fsw	incoming shortwave radiation	W/m^2
fswabs	absorbed shortwave radiation	W/m^2
fswthru	shortwave penetrating to ocean	W/m^2
fswthru_hist	shortwave penetrating to ocean (*fswthru*) for history	W/m^2
fyear	current data year	
fyear_final	last data year	
fyear_init	*initial data year	
G		
gravit	gravitational acceleration	9.80616 m/s^2
grid_file	*input file for grid info	

Name	Description	Default Values
grid_type	*'rectangular' or 'displace_pole' or 'column'	displaced_pole
Gstar	used to compute ridging participation function	0.15
H		
hfrazilmin	minimum thickness of new frazil ice	0.05 m
hi_min	minimum ice thickness for thinnest ice category	0.01 m
hicen	ice thickness in category *n*	m
Hin_max	category limits	m
hist_avg	*if true, write averaged data instead of snapshots	T
histfreq	*history output frequency; y, m, w, d or 1	m
history_dir	*path to history output files	
history_file	*output file for history	
Hm	land/boundary mask, thickness (T-cell)	
hmix	ocean mixed layer depth	20 m
hsnomin	minimum thickness for which T_s is computed	1. x 10^{-6} m
Hstar	determines mean thickness of ridged ice	25. m
HTE	length of eastern edge (Δy) of T-cell	m
HTN	length of northern edge (Δx) of T-cell	m
HTS	length of southern edge (Δx) of T-cell	m
HTW	length of western edge (Δy) of T-cell	m
I		
i0vis	fraction of penetrating visible solar radiation	0.70
icells	number of grid cells with specified property (for vectorization)	
Ice_ref_salinity	reference salinity for ice-ocean exchanges	4. psu
iceruf	ice surface roughness	5. x 10^{-4} m
icetmask	ice extent mask (T-cell)	
iceumask	ice extent mask (U-cell)	
idate	the date at the end of the current time step (yyyymmdd)	
ierr	general-use error flag	
i(j)hi	last *i(j)* index of physical domain (local)	
i(j)lo	first *i(j)* index of physical domain (*local*)	
ilyr1	index of the top layer in each cat (for *eicen*)	

Name	Description	Default Values
ilyrn	index of the bottom layer in each cat (for *eicen*)	
i(j)mt_global	number of physical gridpts in *x(y)* direction, local domain	
i(j)mt_local	total no. of gridpoints in *x(y)* direction, local domain	
Int_kind	definition of an integer	kind(1)
ip, jp	local processor coordinates for writing debugging data	
istep	local step counter for time loop	
istep0	*number of steps taken in previous run	0
istep1	total number of steps at current time step	
K		
kappav	visible extinction coeff. In ice, wavelength < 700 nm	1.4/m
kappan	visible extinction coeff. In ice, wavelength > 700 nm	17.6/m
kcatbound	*category boundary formula	0
kdyn	*type of dynamics (1 = EVP, 0 = off)	1
Kg_to_g	kg to g conversion factor	1000.
kice	thermal conductivity of fresh ice	2.03 W/m/deg
kimin	minimum conductivity of saline ice	W/m/deg
kitd	*type of ITD conversions (1 = delta fxn, 1 = linear remap)	1
kmt_file	*input file for land mask info	
krdg_partic	*ridging participation function	1
krdg_redist	*ridging redistribution function	1
ksmooth	*1 = smooth the ice strength	0
ksno	thermal conductivity of snow	0.30 W/m/deg
kstrength	*ice strength formulation (1 = [13], 0 = [5])	1
L		
l_conservation_check	if true, check conservation	
Lfresh	latent heat of melting of fresh ice = *Lsub=Lvap*	J/kg
lhcoef	transfer coefficient for latent heat	
Log_kind	definition of a logical variable	kind(.true.)

Name	Description	Default Values
Lsub	latent heat of sublimation for fresh water	2.835 x 10^6 J/kg
Lvap	latent heat of vaporization for fresh water	2.501 x 10^6 J/kg
M		
m_to_cm	meters to cm conversion	100.
m1	constant for lateral melt rate	1.6x10^{-6} m/s deg^{-m2}
m2	constant for lateral melt rate	1.36
m2_to_km2	m^2 to km^2 conversion	1 x10^{-6}
mask_n(s)	northern (southern) hemisphere mask	
master_task	task ID for the controlling processor	
mday	day of the month	
meltb	basal ice melt	m
meltl	lateral ice melt	m
meltt	top ice melt	m
melt_onset	day of year that surface melt begins	
month	the month number	
MPI_COMM_ICE	communicator for ice model internal communications (MPI)	
mps_to_cmpdy	m per s to cm per day conversion	8.64 x 10^6
mps_to_compyr	m per s to cm per year conversion	
mtask	local processor number that writes debugging data	
My_task	task ID for the current processor	
N		
nbr_<*dir*>	processor numbers for the N, S, E, W neighbor processors	
ncat	number of ice categories	5
ndte	*number of subcycles	120
ndyn_dt	*number of dynamics/advection steps under thermo	1
new_day	flag for beginning new day	
new_month	flag for beginning new month	
new_week	flag for beginning new week	
new_year	flag for beginning new year	
ngroups	number of groups of flux triangles in remapping	5
nilyr	number of ice layers	4

Name	Description	Default Values
Npt	*total number of time steps (*dt*)	24
ntilay	sum of number of layers in all categories	
ntracer	number of tracers transported in remapping	
Nu_diag	unit number for diagnostics output file	6
Nu_dump	unit number for dump file for restarting	50
Nu_forcing	unit number for forcing data file	49
Nu_grid	unit number for grid file	11
Nu_kmt	unit number for land mask file	12
Nu_nml	unit number for namelist input file	21
Nu_restart	unit number for restart input file	50
Nu_rst_pointer	unit number for pointer to latest restart file	52
num_ghost_cells	no. of rows of ghost cells surrounding each subdomain	1
nyr	year number	
O		
oceanmixed_file	*data file containing ocean forcing data	
oceanmixed_ice	*if true, use internal ocean mixed layer	T
ocn_data_dir	*directory for ocean forcing data	
omega	angular velocity of Earth	7.292×10^{-5} rad/s
one	array of ones which is often useful	1.
opening	rate of ice opening due to divergence and shear	1/s
P		
p001	1/1000	
p01	1/100	
p027	1/36	
p055	1/18	
p1	1/10	
p111	1/9	
p15	15/100	
p166	1/6	
p2	1/5	
p222	2/9	
p25	1/4	

Name	Description	Default Values
p333	1/3	
p4	2/5	
p5	½	
p52083	25/48	
p5625m	-9/16	
p6	3/5	
p666	2/3	
Pi	π	
Pih	π/2	
pi2	2π	
pointer_file	*input file for restarting	
potT	atmospheric potential temperature	K
precip_units	*liquid precipitation data units	
print_global	*if true, print global data	F
print_points	*if true, print point data	F
Pstar	ice strength parameter	2.75×10^4 N/m
puny	a small positive number	1×10^{-11}
Q		
Qa	specific humidity at 10 m	kg/kg
qdp	deep ocean heat flux	W/m^2
qqqice	for saturated specific humidity over ice	1.16378×10^7 kg/m^3
qqqocn	for saturated specific humidity over ocean	6.275724×10^6 kg/m^3
Qref	2 m atmospheric reference specific humidity	kg/kg
R		
Rad_to_deg	degree-radian conversion	180/π
radius	earth radius	6.37×10^6 m
real_kind	definition of single precision real	selected_real_kind(6)
restart	*if true, initialize using restart file instead of defaults	T
restart_dir	*path to restart/dump files	
restore_sst	*restore SST to data	

Name	Description	Default Values
rhoa	air density	kg/m^3
rhofresh	density of fresh water	1000.0 kg/m^3
rhoi	density of ice	917. kg/m^3
rhos	density of snow	330. kg/m^3
rhow	density of seawater	1026 kg/m^3
rnilyr	real(*nlyr*)	
rside	fraction of ice that melts laterally	
S		
saltmax	max salinity, at ice base	3.2 ppm
Sec	seconds elapsed into idate	
secday	number of seconds in a day	86400.
shear	strain rate II component	1/s
shcoef	transfer coefficient for sensible heat	
Sig1(2)	principal stress components (diagnostic)	
sinw	sine of the turning angle in water	0.
snoice	snow-ice formation	m
snowpatch	length scale for parameterizing nonuniform snow coverage	0.02 m
spval	special value (generally over land or undefined regions, in place of 0)	10^30
ss_tltx(y)	sea surface slope in the *x(y)* direction	m/m
Sss	sea surface salinity	psu
Sss_data_type	*source of surface salinity data	
Sst	sea surface temperature	C
Sst_data_type	*source of surface temperature data	
stefan-boltzmann	Stefan-Boltzmann constant	5.67 x 10^{-8} W/m^{2}K^4
stop_now	if 1, end program execution	
strairx(y)	stress on ice by air, in the *x(y)*-direction (centered in U cell)	N/ m^2
strairx(y)T	stress on ice by air, *x(y)*-direction (centered in T cell)	N/ m^2
strength	ice strength (pressure)	N/m
stressp	internal ice stress, $\sigma_{11} + \sigma_{22}$	N/m

Name	Description	Default Values
stressm	internal ice stress, $\sigma_{11} - \sigma_{22}$	N/m
stress12	internal ice stress, σ_{12}	N/m
strintx(y)	divergence of internal ice stress, *x(y)*	N/ m^2
strocnx(y)	ice-ocean stress in the *x(y)*-direction (U-cell)	N/ m^2
strocnx(y)T	ice-ocean stress in the *x(y)*-dir. (T-cell)	N/ m^2
strtlx(y)	surface stress due to sea surface slope	N/ m^2
swv(n)dr(f)	incoming shortwave radiation, visible (near IR), direct (diffuse)	W/ m^2
T		
Tair	air temperature at 10 m	K
tarea	area of T-cell	m^2
tarean	area of northern hemisphere T-cells	m^2
tarear	1/*tarea*	1/ m^2
tareas	area of southern hemisphere T-cells	m^2
Tf	freezing temperature	C
Tffresh	freezing temp of fresh ice	273.15K
time	total elapsed time	s
time_forc	time of last forcing update	s
Timelt	melting temperature of ice top surface	0. C
tinyarea	puny * *tarea*	m^2
TLAT	latitude of cell center	radians
TLON	longitude of cell center	radians
tmask	land/boundary mask, thickness (T-cell)	
tmass	total mass of ice and snow	kg/m^2
Tmin	minimum allowed internal temperature	-100 ° C
Tref	2m atmospheric reference temperature	K
trestore	*SST restoring time scale	days
Tsfc(n)	temperature of ice/snow top surface (in category *n*)	C
Tsf_errmax	max allowed T_{sfc} error (thermodynamics)	5. x 10^{-4} deg
Tsmelt	melting temperature of snow top surface	0. C
TTTice	for saturated specific humidity over ice	5897.8 K
TTTocn	for saturated specific humidity over ocean	5107.4 K

Name	Description	Default Values
U		
uarea	area of U-cell	m^2
uarear	1/*uarea*	
u(v)atm	wind velocity, *x(y)*	m/s
ULAT	latitude of U-cell centers	radians
ULON	longitude of U-cell centers	radians
umask	land/boundary mask, velocity (U-cell)	
umin	min wind speed for turbulent fluxes	1. m/s
u(v)ocn	ocean current, *x(y)* direction	m/s
uvel(vvel)	*x(y)*-component of velocity	m/s
uvm	land/boundary mask, velocity (U-cell)	
V		
vice(n)	volume per unit area of ice (in category *n*)	m
vonkar	von Karman constant	0.4
vsno(n)	volume per unit area of snow (in category *n*)	m
W		
week	week of the year	
wind	wind speed	m/s
work_g1	allocatable, dbl_kind work array	
work_g2	allocatable, dbl_kind work array	
work_gr	allocatable, real_kind work array	
write_history	if true, write history now	
write_ic	if true, write initial conditions now	
work_l1	(imt_local, jmt_local) work array	
work_l2	(imt_local, jmt_local) work array	
work_a	(ilo:ihi, jlo:jhi) work array	
work_b	(ilo:ihi, jlo:jhi) work array	
write_restart	if 1, write restart now	
Y		
ycycle	*number of years in forcing data cycle	
yday	day of the year	

Name	Description	Default Values
year_init	*the initial year	
Z		
zlvl	atmospheric level height	m
zref	reference height for stability	10. m
zTrf	reference height for T_{ref}, Q_{ref}	2. m
zvir	gas constant (water vapor)/gas constant (air) -1	0.606

APPENDIX B

Table of Namelist Options

Name	Type/Options	Description	Default Values / Directory Location
albicei	$0 < \alpha < 1$	near infrared ice albedo for thicker ice	
albicev	$0 < \alpha < 1$	visible ice albedo for thicker ice	
albsnowi	$0 < \alpha < 1$	near infrared, cold snow albedo	
albsnowv	$0 < \alpha < 1$	visible, cold snow albedo	
advection	remap	linear remapping advection	'remap'
	mpdata	2nd order MPDATA	
	upwind	1st order MPDATA	
atm_data_dir	path/	path to atmospheric forcing data directory	
atm_data_type	default	constant values defined in the code	
	ncar	NCAR bulk forcing data	
	LYq	AOMIP/Large-Yeager forcing data	
Dbug	true/false	if true, write atm/ocn data diagnostics	.false.
diag_file	filename	diagnostic output file	
diag_type	stdout	write diagnostic output to stdout	'stdout' (if uncoupled)
	file	write diagnostic output to file	
diagfreq	integer	frequency of diagnostic output in *dt*	24
	eg., 10	once every 10 time steps	
dt	seconds	thermo/transport time step length	3600.
Dump_file	filename prefix	output file for restart dump	'iced'

Name	Type/Options	Description	Default Values / Directory Location
dumpfreq	y, m, d	write restart every *dumpfreq_n* for years, months, days	'y'
dumpfreq_n	integer	frequency restart data is written	
evp_damping	true/false	if true, damp elastic waves [6]	.false.
Fyear_init	yyyy	first year of atmospheric forcing data	
f_<*var*>	true/false	write <*var*> to history	
grid_file	filename	name of grid file to be read	'grid'
grid_type	rectangular, displaced_pole	rectangular: defined in *rectgrid* displaced_pole: read from file in *popgrid*	'displaced_pole'
hist_avg	true/false	write time-averaged data if true write snapshots of data if false	.true.
hist_dir	path/	path to history output directory	
histfreq	y, m, w, d, l	write history output once a year, month, week, day, or every time step	'm'
history_file	filename prefix	output file for history	'iceh'
ice_ic	default	latitude and SST dependent	'default'
	none	no ice	
Istep0	integer	initial time step number	0
kcatbound	0/1	if 0, original category boundary formula if 1, new category boundary formula	0
Kdyn	0 /1	if 0, EVP dynamics OFF if 1, EVP dynamics ON	1
kitd	0 /1	if 0, delta function ITD approx. if 1, linear remapping ITD approx.	1
kmt_file	filename	name of land mask file to be read	'kmt'

Name	Type/Options	Description	Default Values / Directory Location
krdg_partic	0/1	if 0, old ridging participation function if 1, new ridging participation function	1
krdg_redist	0/1	if 0, old ridging redistribution function if 1, new ridging redistribution function	1
kstrength	0 /1	if 0, [5] ice strength formulation if 1, [13] ice strength formulation	1
ndte	integer	number of EVP subcycles	120
Ndyn_dt	integer	number of dynamics/advection/ridging steps per thermo timestep	1
npt	integer	total number of time steps to take	
oceanmixed_file	filename	data file containing ocean forcing data	
oceanmixed_ice	true/false	active ocean mixed layer calculation	.true. (if uncoupled)
ocn_data_dir	path/	path to oceanic forcing data directory	'/scr/posey/pips3/data_in/'
Print_points	true/false	print diagnostic data for two grid points	.false.
precip_units	mm_per_month	liquid precipitation data units	
	mm_per_sec	(default; MKS units)	
Print_global	true/false	print diagnostic data, global sums	.false.
pointer_file	pointer filename	contains restart filename	
restart	true/false	initialize using restart file	.true.
restart_dir	path/	path to restart directory	
restore_sst	true/false	restore SST to data	
sss_data_type	default	constant values defined in the code	
	clim	climatological data	
	ncar	POP ocean forcing data	

Name	Type/Options	Description	Default Values / Directory Location
sst_data_type	default	constant values defined in the code	
	clim	climatological data	
	ncar	POP ocean forcing data	
trestore	integer	SST restoring time scale (days)	
ycycle	integer	no. of years in forcing data cycle	
year_init	yyyy	the initial year, if not using restart	

APPENDIX C

Sample Scripts and Files for PIPS 3.0 Execution

C1 Macros.AIX File

This macros file is used for compiling on the IBM "Babbage" machine at NAVOCEANO.

```
#===============================================================================
========
# CVS $Id: Macros.AIX,v 1.2 2004/02/09 17:55:37 lipscomb Exp $
# CVS $Source: /home/climate/CVS-COSIM/cice/bld/Macros.AIX,v $
# CVS $Name:  $
#===============================================================================
========
# Makefile macros for "romulus" at NAVO
#
# Notes:  (see xlf user's guide for the details)
#  -lmass          => IBM-tuned intrinsic lib
#  -qsmp=noauto    => enable SMP directives, but don't add any
#  -qstrict        => don't turn divides into multiplies, etc
#  -qhot           => higher-order-transformations (eg. loop padding)
#  -qalias=noaryoverlp => assume no array overlap wrt equivalence, etc
#  -qmaxmem=-1     => memory available to compiler during optimization
#  -qipa=level=2   => InterProcedure Analysis (eg. inlining) => slow
compiles
#  -p -pg          => enable profiling (use in both FFLAGS and LDFLAGS)
#  -qreport        => for smp/omp only
#  -bmaxdata:0x80000000 => use maximum allowed data segment size
#  -g              => always leave it on because overhead is minimal (?)
#  -qflttrap=...   => enable default sigtrap (core dump)
#  -C              => runtime array bounds checking (runs slow)
#  -qinitauto=...  => initializes automatic variables
#===============================================================================
========

INCLDIR    := -I. -I/usr/local/include -I/usr/include -
I/usr/lpp/ppe.poe/include \
              -I/site/netcdf/include \
              -I/site/netcdf/include/mod32
SLIBS      :=  -L /usr/local/lib \
               -L /usr/local/lib32/r4i4 -L /site/netcdf/lib
ULIBS      :=
CPP        := /lib/cpp
CPPFLAGS   := -P
CPPDEFS    := -DAIX
#CFLAGS     := -c -O2 -DDISABLE_TIMERS
CFLAGS     := -c -O2
FIXEDFLAGS := -qsuffix=f=f   -qfixed=132
```

```
FREEFLAGS  := -qsuffix=f=f90 -qfree=f90
FC         := mpxlf90_r
FFLAGS     := -O3 -qstrict -qrealsize=8 -qarch=pwr3 -qtune=auto
MOD_SUFFIX := mod
LD         := $(FC)
LDFLAGS    := -bmaxdata:0x80000000

CPPDEFS :=  $(CPPDEFS) -DNPROC_X=$(NX) -DNPROC_Y=$(NY)
#CPPDEFS :=  $(CPPDEFS) -Dfcd_coupled -Dcoupled

ifeq ($(BINTYPE), MPI)
   CPPDEFS :=  $(CPPDEFS) -D_MPI
   SLIBS   :=  $(SLIBS) -lmpi -lnetcdf
Endif
```

C2 Makefile

```
#-------------------------------------------------------------------------
--------
# CVS $Id: Makefile.std,v 1.1 2004/02/09 18:13:52 lipscomb Exp $
# CVS $Source: /home/climate/CVS-COSIM/cice/bld/Makefile.std,v $
# CVS $Name:  $
#-------------------------------------------------------------------------
--------
# Common Makefile: a framework for building all CCSM components and more
#
# Command-line variables
#   MACFILE=<file> ~ the macros definition file to use/include
#   EXEC=<name>    ~ name given to executable, default is a.out
#   VPATH=<vpath>  ~ VPATH            , default is . (cwd only)
#   SRCS=<files>   ~ list of src files, default is all .c .F .F90 files
in VPATH
#   VPFILE=<file>  ~ file with list of dirs, used to create VPATH
#   SRCFILE=<file> ~ file with list of src files, used to create SRCS
#   DEPGEN=<exec>  ~ dependency generator utility, default is makdep
#
#   <macro defns>  ~ any macro definitions found in this file or the
included
#                    MACFILE will be over-ridden by cmd-line macro
definitions
#   MODEL=<model>  ~ a standard macro definition, often found in the
included
#                    MACFILE, used to trigger special compilation flags
#
# Usage examples:
#   % gmake MACFILE=Macros.AIX VPFILE=Filepath MODEL=ccm3 EXEC=atm
#   % gmake MACFILE=Macros.AIX VPFILE=Filepath SRCFILE=Srclist EXEC=pop
#   % gmake MACFILE=Macros.C90 VPATH="dir1 dir2" SRCS="file1.c
file2.F90"
#   % gmake MACFILE=Macros.SUN SRCS="test.F"
```

```
#-----------------------------------------------------------------------
--------

#-----------------------------------------------------------------------
--------
# parse cmd-line and establish values for EXEC, VPATH, SRCS, OBJS, etc
#-----------------------------------------------------------------------
--------

EXEC     := a.out
MACFILE  := NONE
MODEL    := NONE
VPFILE   := NONE
VPATH    := .
SRCFILE  := NONE
SRCS     := NONE
DEPGEN   := ./makdep  # an externally provided dependency generator

ifneq ($(VPATH),.)
  # this variable was specified on cmd line or in an env var
else
  ifneq ($(VPFILE),NONE)
    # explicit list of VPATH dirs is provided
    VPATH := $(wildcard . $(shell cat $(VPFILE) ) )
  endif
endif

ifneq ($(SRCS),NONE)
  # this variable was specified on cmd line or in an env var
else
  ifneq ($(SRCFILE),NONE)
    # explicit list of src files is provided
    SRCS := $(shell cat $(SRCFILE) )
  else
    # list of src files is all .F90 .F .c files in VPATH
    SRCS := $(wildcard $(addsuffix /*.F90 , $(VPATH)) \
                    $(addsuffix /*.[cF], $(VPATH)) )
  endif
endif

OBJS  := $(addsuffix .o, $(sort $(basename $(notdir $(SRCS)))))
DEPS  := $(addsuffix .d, $(sort $(basename $(notdir $(SRCS)))))
INCS  := $(patsubst %,-I%, $(VPATH) )
RM    := rm

.SUFFIXES:
.SUFFIXES: .F90 .F .c .o

all: $(EXEC)
```

```
#-------------------------------------------------------------------------
--------
# include the file that provides macro definitions required by build
rules
# note: the MACFILE may not be needed for certain goals
#-------------------------------------------------------------------------
--------

ifneq ($(MAKECMDGOALS), db_files)
  -include $(MACFILE)
endif

#-------------------------------------------------------------------------
--------
# echo file names, paths, compile flags, etc. used during build
#-------------------------------------------------------------------------
--------

db_files:
     @echo " "
     @echo "* EXEC    := $(EXEC)"
     @echo "* MACFILE := $(MACFILE)"
     @echo "* VPFILE  := $(VPFILE)"
     @echo "* VPATH   := $(VPATH)"
     @echo "* SRCFILE := $(SRCFILE)"
     @echo "* INCS    := $(INCS)"
     @echo "* SRCS    := $(SRCS)"
     @echo "* OBJS    := $(OBJS)"
     @echo "* DEPS    := $(DEPS)"
db_flags:
     @echo " "
     @echo "* cpp     := $(CPP) $(CPPFLAGS) $(CPPDEFS) $(INCS)
$(INCLDIR)"
     @echo "* cc      := cc -c $(CFLAGS) $(INCS) $(INCLDIR)"
     @echo "* .F.o    := $(FC) -c $(FFLAGS) $(FIXEDFLAGS) $(INCS)
$(INCLDIR)"
     @echo "* .F90.o  := $(FC) -c $(FFLAGS) $(FREEFLAGS) $(INCS)
$(INCLDIR)"

#-------------------------------------------------------------------------
--------
# build rules: MACFILE, cmd-line, or env vars must provide the needed
macros
#-------------------------------------------------------------------------
--------

$(EXEC): $(OBJS)
     $(LD) -o $(EXEC) $(LDFLAGS) $(OBJS) $(ULIBS) $(SLIBS)

.c.o:
     cc $(CFLAGS) $(CPPDEFS) $(INCS) $(INCLDIR) $<
```

```
.F.o:
	$(CPP) $(CPPFLAGS) $(CPPDEFS) $(INCS) $(INCLDIR) $< > $*.f
	$(FC) -c $(FFLAGS) $(FIXEDFLAGS) $(INCS) $(INCLDIR) $*.f

.F90.o:
	$(CPP) $(CPPFLAGS) $(CPPDEFS) $(INCS) $(INCLDIR) $< > $*.f90
	$(FC) -c $(FFLAGS) $(FREEFLAGS) $(INCS) $(INCLDIR) $*.f90

mostlyclean:
	$(RM) -f *.f *.f90

clean:
	$(RM) -f *.f *.f90 *.d *.mod *.o
#	$(RM) -f *.f *.f90 *.d *.$(MOD_SUFFIX) $(OBJS)

realclean:
	$(RM) -f *.f *.f90 *.d *.$(MOD_SUFFIX) $(OBJS) $(EXEC)

#---------------------------------------------------------------------------
--------
# Build & include dependency files
#---------------------------------------------------------------------------
--------
# ASSUMPTIONS:
# o an externally provided dependency generator, $(DEPGEN), is
available,
#   its cmd line syntax is compatible with the build rules below.   Eg,
for
#   each .o file, there is a corresponding .d (dependency) file, and
both
#   will be dependent on the same src file, eg.    foo.o foo.d : foo.F90
#   Also, the dependency generator's capabilities, limitations, and
assumptions
#   are understood & accepted.
#---------------------------------------------------------------------------
--------

%.d : %.c
	@ echo "Building dependency for $@"
	@ $(DEPGEN) -f $(INCS) $< | head -3  > $@
%.d : %.F
	@ echo "Building dependency for $@"
	@ $(DEPGEN) -f $(INCS) $<  > $@
%.d : %.F90
	@ echo "Building dependency for $@"
	@ $(DEPGEN) -f $(INCS) $<  > $@
%.d : %.H
	@ echo "Building dependency for $@"
	@ $(DEPGEN) -f $(INCS) $<  > $@

# the if-tests prevent DEPS files from being created when they're not
needed
```

```
ifneq ($(MAKECMDGOALS), db_files)
ifneq ($(MAKECMDGOALS), db_flags)
ifneq ($(MAKECMDGOALS), mostlyclean)
ifneq ($(MAKECMDGOALS), clean)
ifneq ($(MAKECMDGOALS), realclean)
    -include $(DEPS)
endif
endif
endif
endif
endif
```

C3 Ice_in Input Parameter File

```
&ice_nml
    year_init    = 0001
  , istep0       = 0
  , dt           = 2880.0
  , ndte         = 120
  , npt          = 70
  , diagfreq     = 30
  , histfreq     = 'h'
  , dumpfreq     = 'd'
  , dumpfreq_n   = 1
  , hist_avg     = .false.
  , restart      = .true.
  , print_points = .true.
  , print_global = .true.
  , kitd         = 1
  , kcatbound    = 1
  , kdyn         = 1
  , kstrength    = 1
  , krdg_partic  = 1
  , krdg_redist  = 0
  , evp_damping  = .false.
  , advection    = 'remap'
  , grid_type    = 'pips'
  , grid_file    = 'grid_cice_1280x720.r'
  , kmt_file     = 'kmt'
  , dump_file    = 'iced'
  , restart_dir  = '/scr/posey/pips3/'
  , pointer_file = '/scr/posey/pips3/ice.restart_file'
  , history_dir  = '/scr/posey/pips3/'
  , history_file = 'iceh'
  , diag_file    = 'ice_diag.d'
  , oceanmixed_ice = .true.
  , albicev      = 0.65
  , albicei      = 0.65
  , albsnowv     = 0.85
  , albsnowi     = 0.85
  , ycycle       = 1
```

```
  , fyear_init   =  2008
  , atm_data_dir = '/scr/posey/pips3/data_in/'
  , ocn_data_dir = '/scr/posey/pips3/data_in/'
/

&icefields_nml
    f_hi        = .true.
  , f_hs        = .true.
  , f_Tsfc      = .false.
  , f_aice      = .true.
  , f_uvel      = .true.
  , f_vvel      = .true.
  , f_fswdn     = .false.
  , f_flwdn     = .false.
  , f_snow      = .false.
  , f_snow_ai   = .false.
  , f_rain      = .false.
  , f_rain_ai   = .false.
  , f_sst       = .true.
  , f_sss       = .true.
  , f_uocn      = .true.
  , f_vocn      = .true.
  , f_frzmlt    = .false.
  , f_fswabs    = .false.
  , f_fswabs_ai = .false.
  , f_albsni    = .false.
  , f_alvdr     = .false.
  , f_alidr     = .false.
  , f_flat      = .false.
  , f_flat_ai   = .false.
  , f_fsens     = .false.
  , f_fsens_ai  = .false.
  , f_flwup     = .false.
  , f_flwup_ai  = .false.
  , f_evap      = .false.
  , f_evap_ai   = .false.
  , f_Tref      = .false.
  , f_Qref      = .false.
  , f_congel    = .false.
  , f_frazil    = .false.
  , f_snoice    = .false.
  , f_meltt     = .false.
  , f_meltb     = .false.
  , f_meltl     = .false.
  , f_fresh     = .false.
  , f_fresh_ai  = .false.
  , f_fsalt     = .false.
  , f_fsalt_ai  = .false
  , f_fhnet     = .false.
  , f_fhnet_ai  = .true.
  , f_fswthru   = .false.
```

```
  , f_fswthru_ai= .false.
  , f_strairx   = .false.
  , f_strairy   = .false.
  , f_strtltx   = .false.
  , f_strtlty   = .false.
  , f_strcorx   = .false.
  , f_strcory   = .false.
  , f_strocnx   = .true.
  , f_strocny   = .true.
  , f_strintx   = .true.
  , f_strinty   = .true.
  , f_strength  = .true.
  , f_opening   = .false.
  , f_divu      = .false.
  , f_shear     = .false.
  , f_sig1      = .false.
  , f_sig2      = .false.
  , f_dvidtt    = .false.
  , f_dvidtd    = .false.
  , f_daidtt    = .false.
  , f_daidtd    = .false.
  , f_mlt_onset = .false.
  , f_frz_onset = .false.
  , f_dardg1dt  = .false.
  , f_dardg2dt  = .false.
  , f_dvirdgdt  = .false.
  , f_hisnap    = .false.
  , f_aisnap    = .false.
  , f_aice1     = .true.
  , f_aice2     = .true.
  , f_aice3     = .true.
  , f_aice4     = .true.
  , f_aice5     = .true.
  , f_aice6     = .false.
  , f_aice7     = .false.
  , f_aice8     = .false.
  , f_aice9     = .false.
  , f_aice10    = .false.
  , f_vice1     = .true.
  , f_vice2     = .true.
  , f_vice3     = .true.
  , f_vice4     = .true.
  , f_vice5     = .true.
  , f_vice6     = .false.
  , f_vice7     = .false.
  , f_vice8     = .false.
  , f_vice9     = .false.
  , f_vice10    = .false.
/
```

C4 Clean_ice script

```
#! /bin/csh -f

#setenv SYSTEM_USERDIR /pvfs/lipscomb  # ORNL defines this automatically

setenv SRCDIR $HOME/cice.v3.1/cice

setenv CBLD   $SRCDIR/bld
setenv EXEDIR $SYSTEM_USERDIR/rundir
setenv OBJDIR $EXEDIR/compile

cd $OBJDIR
pwd

# Clean compile directory
gmake -f $CBLD/Makefile clean  || exit 2

cd $EXEDIR
```

APPENDIX D

Sample Scripts for PIPS 3.0/NCOM Coupled Execution

After the global NCOM (GNCOM) runs each day, it calls a regional post-processing script (*launch_pips.com*) which runs *ncom2pips.com* for the current day. *Ncom2pips.com* uses the MODAS regridding routines to regrid the GNCOM surface temperature and ocean velocities onto the PIPS grid in a netCDF format. A program (average_file_realtime.f) then takes the 3 hourly outputs from GNCOM and a makes a daily mean for input into the ice model. These GNCOM fields (SST and ocean currents) are read directly into PIPS and used to make a 96 hour (4-day) forecast.

After PIPS runs each day, it calls a post_processing script (*launch_ncom.com*) which runs *pips2ncom.com*. This script launches a parallel process, *regrid_pips_poe.com* of serial jobs *regrid_pips.com* to regrid the PIPS output every 3 hours onto the GNCOM grid. The ice concentration, heat flux and ice-ocean stresses are regridded in this manner. The SST files are only regridded at the 00 hour. The *fill4.com* routine masks noise at the land-sea boundaries and then *landmask.com* scripts puts the GCNOM landmask onto the newly gridded fields.

The ice-ocean stresses, heat fluxes and ice concentration are brought into ncom_4.0/bin/sigz.global/ncom_nc2atmice.exe which makes an "atmice.A" file. The PIPS fields are used through hour 96 and then persisted for an additional 24 hours. The "atmice.A" file is saved for the following day.

The following day, ncom_4.0/bin/sigz.global/ncom_atmuvhtms_icemask3.01.exe is run after the NOGAPS forcing OSFLX_1.A is made. A check is then made to the NOGAPS forcing field - when PIPS forecasts indicate that ice is present (concentration > 1%):

1) the PIPS ice-ocean stresses replaces the NOGAPS wind stresses
2) the PIPS heat flux replaces the GNCOM bulk-formulae heat fluxes

The SST outputs are blended into the MODAS 2D synthetics, by using PIPS ice concentration to determine where there is ice coverage. A new ice mask (blendmask) is made and applied to the PIPS SST netCDF output. An inverse ice mask (blendmaski) is also made and applied to the MODAS sstf output. These two fields are added together using *grdmath.com* in the GNCOM synthetic pre-processing to be used the following day. Persisted fields are used to make up the last 24 hours.

The cycle is repeated daily.

D1 launch_pips.com

```
#!/bin/csh
#
# script to make PIPS3 inputs
#  Programmer: Lucy F. Smedstad, NRL Code 7323
#     12 February 2008
#
set echo
set verbose
goto START

START:

#
if (!($?NSCRIPTS)) setenv NSCRIPTS /u/home/ooc/models/ncom1/scripts
set unflaggedargs = (idtg1analysis idtg1start idtg1end)
source $NSCRIPTS/arg_eval.com
source NCOM.env
/bin/cp $HOMD/NCOM.env .
/bin/cp $HOMD/NCOM.env /scr/ooc/data/ncom1/glb8_3b/work/pips3
source NCOM.env
cd $PIPSHOMD

#

set c =
/scr/ooc/data/ncom1/pips/pipsfrcst/ncom2pips_${idtg1analysis}.com
awk -f today.awk idtg1analysis=$idtg1analysis ncom2pips.com >! $c

 bsub < $c

DONE:
exit
ERROR:
echo  $idtg1analysis 'not run for NCOM'
exit 1
```

D2 ncom2pips.com

```
#! /bin/csh
#BSUB -a ncom2pips
#BSUB -o /scr/ooc/data/ncom1/glb8_3b/work/pips3/ncom2pips.log
#BSUB -e /scr/ooc/data/ncom1/glb8_3b/work/pips3/ncom2pips.log
#BSUB -P NAVOSOOC        # Charging project or group name.
#BSUB -W 01:35           # Wall clock time of 35 minutes.
#BSUB -q internal   # Queue name.
#BSUB -n 1              # Number of processors or total tasks.
#BSUB -R "span[ptile=8]"

set echo
set verbose
set idtg2s = ( 00 03 06 09 12 15 18 21 )
set idtg2sf = ( 000 003 006 009 012 015 018 021 024 027 030 033 036 039
042 045 048 051 054 057 060 063 066 069 072 075 078 081 084 087 090 093
096 )
#if ($#argv > 0) then
#  set idtg1start = $1
#else
#  set idtg1start = `date +%Y%m%d`
#endif
# set idtg1analysis = IDTG1ANALYSIS
source NCOM.env
set i = `$BIN_MODAS/addndays yyyymmdd $idtg1analysis -1`

setenv XLFRTEOPTS namelist=old

while ($i < $idtg1end)

  set i = `$BIN_MODAS/addndays yyyymmdd $i 1`
  if ($i < $idtg1end) then
    foreach idtg2 ($idtg2sf)
      echo $MODAS2HOME
      setenv | grep MODAS2
      ${NSCRIPTS}/regrid_ncomtostandard.com
$NCDIR/sst_${runname}_${i}00_t${idtg2}h.nc
$PIPSDIR/sst_ncom_${i}00_t${idtg2}h.nc -NCDIRMASTER $NCDIRMASTER -
filexout $PIPSNCMASTER/model_lon.nc -fileyout $PIPSNCMASTER/model_lat.nc
-AREA none -DEPTHS none
#      ${NSCRIPTS}/regrid_ncomtostandard.com
$NCDIR/sss_${runname}_${i}${idtg2}.nc $PIPSDIR/sss_ncom_${i}${idtg2}.nc
-NCDIRMASTER $NCDIRMASTER -filexout $PIPSNCMASTER/model_lon.nc -fileyout
$PIPSNCMASTER/model_lat.nc   -AREA none -DEPTHS none
      ${NSCRIPTS}/regriduv_ncomtostandard.com
$NCDIR/ssu_${runname}_${i}00_t${idtg2}h.nc
$NCDIR/ssv_${runname}_${i}00_t${idtg2}h.nc
```

```
$PIPSDIR/ssu_ncom_${i}${idtg2}us.nc $PIPSDIR/ssv_ncom_${i}${idtg2}us.nc
-NCDIRMASTER $NCDIRMASTER -filexout $PIPSNCMASTER/model_lon.nc -fileyout
$PIPSNCMASTER/model_lat.nc -AREA none -DEPTHS none -gridtypein c -
gridtypeout a
$PIPSHOMD/do_smooth.com $PIPSDIR/ssu_ncom_${i}${idtg2}us.nc
$PIPSDIR/ssu_ncom_${i}00_t${idtg2}h.nc 9
$PIPSHOMD/do_smooth.com $PIPSDIR/ssv_ncom_${i}${idtg2}us.nc
$PIPSDIR/ssv_ncom_${i}00_t${idtg2}h.nc 9
    end
 endif
#
end

chdir  $PIPSHOMD
awk -f tmp.awk tmp=$idtg1analysis \
  pips3_tmp.lsf >! pips3_run.lsf
#
bsub < pips3_run.lsf
```

D3 launch_ncom.com

```
#!/bin/csh
#
#  script to get NCOM inputs
#  modification of lsf script
#  by Chris DeHaan modification of script by:
#  Programmers: Lucy F. Smedstad and Charlie N. Barron, NRL Code 7323
#     21 May 2004
#
set echo
set verbose
goto START

START:

#
if (!($?NSCRIPTS)) setenv NSCRIPTS /u/home/ooc/models/ncom1/scripts
set unflaggedargs = (idtg1analysis idtg1start idtg1end)
source $NSCRIPTS/arg_eval.com
set HOMD = /u/home/ooc/models/ncom1/glb8_3b
echo $idtg1analysis
/bin/cp $HOMD/NCOM.env .
source NCOM.env

#
cd $PIPSHOMD
/bin/rm $PIPSDIR_W/pips2ncom.log
```

```
set c = $PIPSDIR_W/pips2ncom_${idtg1analysis}.com
awk -f today.awk idtg1analysis=$idtg1analysis pips2ncom.com >! $c

bsub < $c

DONE:
exit
ERROR:
exit 1
```

D4 pips2ncom.com

```
#!/bin/csh
#BSUB -a pips2ncom
#BSUB -o /scr/ooc/data/ncom1/pips/pipsfrcst/pips2ncom.log
#BSUB -e /scr/ooc/data/ncom1/pips/pipsfrcst/pips2ncom.log
#BSUB -P NAVOSOOC        # Charging project or group name.
#BSUB -W 01:55           # Wall clock time of 55 minutes.
#BSUB -q internal       # Queue name.
#BSUB -n 1              # Number of processors or total tasks.
#BSUB -R "span[ptile=8]"

# 1a. make sure BIN and NSCRIPTS are defined correctly

set echo
set verbose
# set idtg1analysis = IDTG1ANALYSIS
source NCOM.env
/bin/cp $NCOMHOMD/NCOM.env $PIPSDIR_W
/bin/cp $NCOMHOMD/NCOM.env .

#if ($#argv > 0) then
#  set idtg1analysis = $1
#else
#  set idtg1analysis = `date +%Y%m%d`
#endif
# set idtg1analysis = IDTG1ANALYSIS
set mm = `echo $idtg1analysis | awk '{a=$1;b=substr(a,5,2);print b}'`
set dd = `echo $idtg1analysis | awk '{a=$1;b=substr(a,7,2);print b}'`
set pipsend = `$BIN_NCOM/addndays yyyymmdd $idtg1end -1`
set pipsstart = `$BIN_NCOM/addndays yyyymmdd $idtg1start 2`
set i = `$BIN_NCOM/addndays yyyymmdd $idtg1start -1`
set nrec = 0
chdir $PIPSDIR_W
touch newpips_$$.lis
/bin/rm RUNNING_*
foreach p (aice ustr vstr hflx)
  /bin/rm ${p}??.nc
```

```
end
while ($i < $pipsstart)
   set i = `$BIN_NCOM/addndays yyyymmdd $i 1`
   foreach tt ( $outtimes)
     @ nrec = 1 + $nrec
     set nn = `echo $nrec | awk '{printf "%02.2d\n",$1}'`
     if (-e strocnx_ncom_${i}${tt}fm.nc) ln -fs
strocnx_ncom_${i}${tt}fm.nc ustr${nn}.nc
     if (-e strocny_ncom_${i}${tt}fm.nc) ln -fs
strocny_ncom_${i}${tt}fm.nc vstr${nn}.nc
     if (-e fhnet_ai_ncom_${i}${tt}fm.nc) ln -fs
fhnet_ai_ncom_${i}${tt}fm.nc hflx${nn}.nc
     if (-e aice_ncom_${i}${tt}fm.nc) ln -fs aice_ncom_${i}${tt}fm.nc
aice${nn}.nc
   end
end
set i = `$BIN_NCOM/addndays yyyymmdd $pipsstart -1`
while ($i < $pipsend)
  set i = `$BIN_NCOM/addndays yyyymmdd $i 1`
  set yyyy = `echo $i | awk '{a=$1;b=substr(a,1,4);print b}'`
  set mm = `echo $i | awk '{a=$1;b=substr(a,5,2);print b}'`
  set dd = `echo $i | awk '{a=$1;b=substr(a,7,2);print b}'`
  set PIPSyeardiff = 0
@ PIPSyeardiff = $PIPSyeardiff + $yyyy - $PIPSstartyear
   set PIPSYEAR = 00$PIPSyeardiff
   set n = 0
  foreach pfile ($PIPS/iceh.${PIPSYEAR}-${mm}-${dd}*.nc)
    @ tthour = $n * $PIPSHRINC
    set tt = `echo $tthour | awk '{printf "%02.2d\n",$1}'`
    @ n = $n + 1
    ln -sf $pfile iceh_${i}${tt}.nc
    echo ${i}${tt} >> newpips_$$.lis
  end
end
set c = regrid_pips_${idtg1analysis}.com

  set jn = `echo $c | sed -e 's/.com$//'`
  if (-e $c) /bin/rm $c
  echo "#\!/bin/csh" >! $c
  echo "#BSUB -J $jn" >> $c
  echo "#BSUB -o $PIPSDIR_W/${jn}.log" >> $c
  echo "#BSUB -e $PIPSDIR_W/${jn}.log" >> $c
  echo "#BSUB -n $PIPSprocs" >> $c
  echo "#BSUB -a poe" >> $c
  echo "#BSUB -W 1:55" >> $c
  echo "#BSUB -P NAVOSOOC" >> $c
  echo "#BSUB -q internal" >> $c
  echo "#BSUB -R 'span[ptile=$tasks_allowed_per_node]'" >> $c
  echo "#" >> $c
  echo "setenv MP_PROCS $PIPSprocs" >> $c
  echo "chdir $PIPSDIR_W" >> $c
```

```
  echo "mpirun.lsf $PIPSHOMD/regrid_pips_poe.com
$PIPSDIR_W/newpips_$$.lis 1 -idtg1analysis $idtg1analysis -pulse 0" >>
$c
  echo "wait" >> $c
  bsub < $c
DONE:
exit
ERROR:
exit 1
```

D5 regrid_pips_poe.com

```
#!/bin/csh -f
#
# script to run regrid_pips jobs under in parallel
#  Lucy F. Smedstad
#  NRL Code 7323
#  14 September 2007
#
cd /u/home/$user/models/ncom1/pips3
source NCOM.env
if ($#argv > 0) then
  set daylist = $PIPSDIR_W/$1
  if (!(-e $daylist)) then
    set daylist = $1
  endif
  if (!(-e $daylist)) then
    set daylist = $PIPSDIR_W/regrid_pips_days.lis
  endif
endif
if (!(-e $daylist)) then
  echo file $daylist not found
  goto ERROR
endif
set nidtgs = `cat $daylist | wc -l`
# note that $MP_CHILD is a number from zero to the total processors -1
set mynode = $MP_CHILD
@ idate = $mynode + 1
#
set i = $idate
#while ($i > $tasks_3d_used_per_node)
#  if ($i <= $tasks_allowed_per_node) then
#    goto DONE
#  endif
#  @ i = $i - $tasks_allowed_per_node
#  @ idleprocs_per_node = $tasks_allowed_per_node -
$tasks_3d_used_per_node
#  @ idate = $idate - $idleprocs_per_node
```

```
#end
if ($idate > $nidtgs) then
  goto DONE
else
  set i = `cat $daylist | head -$idate | tail -1`
endif

  #

set c = ${PIPSDIR_W}/regrid_pips_${i}.com
set l = ${PIPSDIR_W}/regrid_pips_${i}.log
if (-e $l) /bin/rm $l
echo "#" >! $c
echo "set echo" >> $c
echo "set verbose" >> $c
echo "echo MP_CHILD $MP_CHILD" >> $c
echo "chdir $PIPSDIR_W" >> $c
echo "set idtg1 = $i" >> $c
cat regrid_pips.com >> $c
chmod 755 $c
csh -f $c >&! $l
if ($status != 0) then
  echo ERROR in $c
  goto ERROR
endif

DONE:
exit
ERROR:
exit 1
```

D6 regrid_pips.com

```
set echo
set verbose
set tt =  `echo $idtg1 | awk '{a=$1;b=substr(a,9,2);print b}'`
set i8 =  `echo $idtg1 | awk '{a=$1;b=substr(a,1,8);print b}'`
source NCOM.env
chdir $PIPSDIR_W
touch RUNNING_${idtg1}
    $BIN_NCOM/extractfrompips iceh_${idtg1}.nc -idtg1in $idtg1
    $BIN_NCOM/extractfrompips iceh_${idtg1}.nc -idtg1in $idtg1 -f
fhnet_ai
    if ($tt == '00') then
       $BIN_NCOM/extractfrompips iceh_${idtg1}.nc -idtg1in $idtg1 -f sst
       set infile = $PIPSDIR_W/sst_iceh_${idtg1}.nc
       set outfile = pipssst_${idtg1}.nc
```

```
       set maskfile = $PIPSHOMD/pips_mask.nc
       set landmaskfile = $PIPSHOMD/pips_landmask.nc
       $PIPSHOMD/grdmath.com $infile $maskfile out_$$.nc mask 0 0
       $PIPSHOMD/fill4.com out_$$.nc outf_$$.nc
       $PIPSHOMD/grdmath.com outf_$$.nc $landmaskfile out2_$$.nc mask 0
0
       $PIPSHOMD/grdmath.com out2_$$.nc none out_$$.nc maskbull 2.0 0
       $PIPSHOMD/fill4.com out_$$.nc  out2f_$$.nc
       $PIPSHOMD/grdmath.com out2f_$$.nc $landmaskfile $outfile mask 0 0
#       /bin/rm out_$$.nc out2_$$.nc
       ${LSCRIPTS}/regrid_ncomtostandard.com pipssst_${idtg1}.nc
$SYNWORK/sst_pipsmodas_${idtg1}.nc -NCDIRMASTER $PIPSNCMASTER  -AREA
$SYNCOM/region.area -DEPTHS none -gridtypein $gridpips -gridtypeout
$gridpips
    endif

    foreach field ( aice )
       set infile = ${field}_iceh_${idtg1}.nc
       set outfile = fixed_${infile}
       set maskfile = $PIPSHOMD/pips_mask.nc
       set landmaskfile = $PIPSHOMD/pips_landmask.nc
       $PIPSHOMD/grdmath.com $infile $maskfile out_$$.nc mask 0 0
       $PIPSHOMD/fill4.com out_$$.nc  outf_$$.nc
       $PIPSHOMD/grdmath.com outf_$$.nc $landmaskfile out2_$$.nc mask 0
0
       $PIPSHOMD/grdmath.com out2_$$.nc none out_$$.nc maskbull 2.0 0
       $PIPSHOMD/fill4.com out_$$.nc  out2f_$$.nc
       $PIPSHOMD/grdmath.com out2f_$$.nc $landmaskfile $outfile mask 0 0
#       /bin/rm out_$$.nc out2_$$.nc
     end
   ${LSCRIPTS}/regrid_ncomtostandard.com fixed_aice_iceh_${idtg1}.nc
aice_pipsmodas_${idtg1}.nc -NCDIRMASTER $PIPSNCMASTER  -AREA
$SYNCOM/region.area -DEPTHS none -gridtypein $gridpips -gridtypeout
$gridpips
NEWSST:
       # put special values in areas with very little or no ice
       $PIPSHOMD/grdmath.com aice_pipsmodas_${idtg1}.nc none
aicesp_$$.nc min 0.0001 0
       # put ones in areas with some ice
       $PIPSHOMD/grdmath.com aicesp_$$.nc none aice1sp_$$.nc linear 1 0
       # expand out to define with ones areas very near or under ice
       $PIPSHOMD/fillcreep.com aice1sp_$$.nc aicenear1sp_$$.nc 6
       # expand out to define with ones areas somewhat near or under ice
       $PIPSHOMD/fillcreep.com aicenear1sp_$$.nc aicenear2sp_$$.nc 16
       # change open water special values to zeros
       $PIPSHOMD/grdmath.com aicenear2sp_$$.nc none aicenear2_$$.nc
fspval 0 0
       # put special values in areas somewhat near ice
       $PIPSHOMD/grdmath.com aicenear2_$$.nc none aicefar0sp_$$.nc max
0.0001 0
       # make a blend mask with ones under or very near ice, zeros far
from ice, spval in between
```

```
   $PIPSHOMD/grdmath.com aicenear1sp_$$.nc aicefar0sp_$$.nc
preblendmask_$$.nc fill  0 0
       # blend between very near ice 1 and far from ice 0
       $PIPSHOMD/fillsq.com preblendmask_$$.nc blendmask_$$.nc
       # invert mask to have 1 far from ice, 0 near
       $PIPSHOMD/grdmath.com blendmask_$$.nc none blendmaski_${idtg1}.nc
linear 1 -1
       # fill to have cice sst values in regions somewhat near ice
       $PIPSHOMD/fillcreep.com $SYNWORK/sst_pipsmodas_${idtg1}.nc
sst_pipsmodasf_$$.nc 27
       # replace cice spval with large number to help spot errors in
masks
       $PIPSHOMD/grdmath.com sst_pipsmodasf_$$.nc none
sst_pipstest_$$.nc fspval 1000000 0
       # multiply cice sst times cice blend mask
       $PIPSHOMD/grdmath.com sst_pipstest_$$.nc blendmask_$$.nc
$SYNWORK/sst_pipsblend_${idtg1}.nc product 1 0



    ${LSCRIPTS}/regriduv_ncomtostandard.com strocnx_iceh_${idtg1}.nc
strocny_iceh_${idtg1}.nc strocnx_ncom_${idtg1}.nc
strocny_ncom_${idtg1}.nc -NCDIRMASTER $PIPSNCMASTER -filexout
$NCDIRMASTER/model_lon.nc -fileyout $NCDIRMASTER/model_lat.nc -AREA none
-DEPTHS none -fileangout $NCDIRMASTER/model_ang.nc -gridtypein $gridpips
-gridtypeout $gridpips
    ${LSCRIPTS}/regrid_ncomtostandard.com fixed_aice_iceh_${idtg1}.nc
aice_ncom_${idtg1}.nc -NCDIRMASTER $PIPSNCMASTER -filexout
$NCDIRMASTER/model_lon.nc -fileyout $NCDIRMASTER/model_lat.nc -AREA none
-DEPTHS none -gridtypein $gridpips -gridtypeout $gridpips -fileangout
$NCDIRMASTER/model_ang.nc
    ${LSCRIPTS}/regrid_ncomtostandard.com fhnet_ai_iceh_${idtg1}.nc
fhnet_ai_ncom_${idtg1}.nc -NCDIRMASTER $PIPSNCMASTER -filexout
$NCDIRMASTER/model_lon.nc -fileyout $NCDIRMASTER/model_lat.nc -AREA none
-DEPTHS none -gridtypein $gridpips -gridtypeout $gridpips -fileangout
$NCDIRMASTER/model_ang.nc

    foreach d ( x y )
       $PIPSHOMD/landmask.com strocn${d}_ncom_${idtg1}.nc
$NCDIRMASTER/model_h.nc
      $PIPSHOMD/fill.com strocn${d}_ncom_${idtg1}m.nc
strocn${d}_ncom_${idtg1}f.nc
       $PIPSHOMD/landmask.com strocn${d}_ncom_${idtg1}f.nc
$NCDIRMASTER/model_h.nc
    end

    $PIPSHOMD/landmask.com aice_ncom_${idtg1}.nc $NCDIRMASTER/model_h.nc
    $PIPSHOMD/fill.com aice_ncom_${idtg1}m.nc aice_ncom_${idtg1}f.nc
    $PIPSHOMD/landmask.com aice_ncom_${idtg1}f.nc
$NCDIRMASTER/model_h.nc
    $PIPSHOMD/landmask.com fhnet_ai_ncom_${idtg1}.nc
$NCDIRMASTER/model_h.nc
    $PIPSHOMD/fill.com fhnet_ai_ncom_${idtg1}m.nc
fhnet_ai_ncom_${idtg1}f.nc
    $PIPSHOMD/landmask.com fhnet_ai_ncom_${idtg1}f.nc
$NCDIRMASTER/model_h.nc
```

```
/bin/rm RUNNING_${idtg1}
if ($MP_CHILD == 0) then
      @ ntries = 0
      set i = 1
    while ( $i > 1 )
       @ ntries = $ntries + 1
      if ($ntries > 100) then
         echo ERROR:  regrid unsucessful
         goto ERROR
      endif
      sleep 60
      ls RUNNING_* > regrid_done
      set i = `cat regrid_done | wc -l`
      /bin/rm regrid_done
    end
  set echo
  set verbose
  set idtg1analysis = `$BIN_NCOM/addndays yyyymmdd $i8 1`
  echo $idtg1analysis
  set c = $PIPSDIR_W/remake_ncom_inputs_${idtg1analysis}.com
  awk -f $PIPSHOMD/today.awk idtg1analysis=$idtg1analysis
$PIPSHOMD/remake_ncom_inputs.com >! $c
  csh $c
endif
DONE:
exit 0
```

D7 fill4.com

```
#!/bin/csh -f
goto START
USAGE:
echo 'usage fill.com infile.nc [outfile.nc]'
echo 'default outfile is replacement of infile.nc with filled version'
echo fill uses creepbotfill
goto ERROR

START:

if ($#argv < 1) goto USAGE

set unflaggedargs = (infile outfile)
source NCOM.env
source $NSCRIPTS/arg_eval.com
set infile = $1
set outfile = $2
```

```
if (!(-e $infile)) goto USAGE
if (!($?outfile)) set outfile = $infile
if ($outfile == $infile) then
  set overwrite = 1
  set outfile = t_$$.nc
else
  set overwrite = 0
endif


setenv BIN_MODAS /u/home/ooc/models/MODAS2/bin
setenv XLFRTEOPTS namelist=old
set ncreep = 4
#
#  if isds > 0 then only the isds data set is processed
#  if isds <=0  then the first, last, and increment data sets
#    are given by itlsds, lstsds, incsds
#
#-------------------------------------------------------------------------
-------
#
time $BIN_MODAS/fill << END
 &inputs
   filein  = '$infile'
   fileout = '$outfile'
   isds    =  1
   itlsds   = 1
   lstsds   = 1
   incsds   = 1
   creeping  = t
   itermax = 5
   ncreep = $ncreep
 &end
END
if ( $status != 0 ) then
  echo " ** Error in GDEMGRD.  FATAL ERROR, EXITING SCRIPT ** "
endif

if ($overwrite == 1) /bin/mv $outfile $infile

#
DONE:
exit
ERROR:
echo ERROR in fill.com $0
exit 1
```

D8 landmask.com

```
#!/bin/csh
#
# script to mask one netcdf file with another
#
# programmer: Charlie Barron, NRL code 7323
# Date: 17 July 2001
#
setenv BIN_MODAS /u/home/ooc/models/MODAS2/bin
setenv BIN_MODAS /u/home/fitzgrld/models/modas2/bin
#
if ($#argv < 2) then
  echo USAGE landmask.com datafile.nc maskfile.nc [maskeddatafile.nc]
[mode]
  echo default output is named "datafile"m.nc where input is
"datafile".nc
  echo note that to set mode 4 arguments are required
  echo the maskfile is assumed to indicate land with nonnegative values
  goto ERROR
endif

set datafile = $1
set maskfile = $2

if (!(-e $datafile)) then
  echo ERROR: data file $datafile does not exist
  goto ERROR
endif

if (!(-e $maskfile)) then
  echo ERROR: mask file $maskfile does not exist
  goto ERROR
endif

set samename = 0
set mode = 0
if ($#argv > 2) then
  set maskeddatafile = $3
  if ($maskeddatafile == $datafile) then
    set samename = 1
    set maskeddatafile = tmp_$$.nc
  endif
  if ($#argv > 3) then
    set mode = $4
  endif
else
  set maskeddatafile = `echo $datafile | sed -e 's/.nc$//' -e
's/$/m.nc/'`
endif
```

```
setenv XLFRTEOPTS namelist=old
$BIN_MODAS/botmask << END
 &inputs
   filein    = '$datafile'
   filebotin = '$maskfile'
   fileout   = '$maskeddatafile'
   npastbot  = 0
   mode      = $mode
 &end
END
if ( $status != 0 ) then
  echo " ** Error in botmask.  FATAL ERROR, EXITING SCRIPT ** "
  goto ERROR
else
  if ($samename == 1) then
    /bin/mv $maskeddatafile $datafile
    set maskeddatafile = $datafile
  endif
  echo masked data file is stored in $maskeddatafile
endif
#  done
DONE:
exit
ERROR:
echo ERROR in landmask.com
exit 1
```

D9 grdmath.com

```
#!/bin/csh -f
#
# script to simplify calls to grdmath
# Programmer: Charlie Barron, NRL code 7323
# 15 Jun 2004
#
source NCOM.env
setenv XLFRTEOPTS namelist=old
set LBIN = /u/home/fitzgrld/models/modas2/bin

set echo
set verbose
set minarg = 6
if ($#argv < $minarg) then
  echo 'USAGE: grdmath.com fileain filebin filecout operation vala valb
[...]'
  goto DONE
endif
```

```
set fileain = $1
set filebin = $2
set filecout = $3
set operation = "$4"
set vala = $5
set valb = $6
set idtg1 = 0
set idtg2 = 0
set hh = 0

source $NSCRIPTS/arg_eval.com
if (!($?quiet)) setenv quiet 1

if (!($?fileain)) then
  echo ERROR: fileain is not defined
  goto ERROR
endif
if (!(-e $fileain)) then
  echo ERROR: fileain $fileain is not found
  goto ERROR
endif
if (!($?filebin)) set filebin = none
if ($?nfiles) then
  if ($nfiles == 1) set filebin = none
endif
if ($filebin != 'none') then
  if (!(-e $filebin)) then
    echo ERROR: file b $filebin not found
    goto ERROR
  else
    set nfiles = 2
  endif
else
  set nfiles = 1
endif

if (!($?filecout)) then
  echo ERROR: output file $filecout not defined
  goto ERROR
endif
set samename = 0
set filefinalout = $filecout
if ($filecout == $fileain) then
  set filecout = cout_tmp_$$.nc
  set samename = 1
endif
if ($filecout == $filebin) then
  set filecout = cout_tmp_$$.nc
  set samename = 1
endif
```

```
if (!($?vala)) set vala = 0.
if (!($?valb)) set valb = 0.

if (!($?operation)) then
  echo ERROR: grdmath operation is not defined
  goto ERROR
endif

if (!($?zmtoft)) set zmtoft = f
if (!($?newdlabel)) set newdlabel = none
if (!($?newdunit)) set newdunit = none
if (!($?newdfmt)) set newdfmt = none
set idtg2 = `echo $idtg2 $hh | awk '{i=$1;h=$2;if
(i<=0){i=1000000*h};printf "%08.8d\n",i}'`

$LBIN/grdmath << END
 &dinput
   fileain   = '$fileain'
   filebin   = '$filebin'
   filecout  = '$filecout'
   nfiles    = $nfiles
   vala      = $vala
   valb      = $valb
   operation = '$operation'
   zmtoft    = $zmtoft
   newdlabel = '$newdlabel'
   newdunit  = '$newdunit'
   newdfmt   = '$newdfmt'
   idtg(1)   = $idtg1
   idtg(2)   = $idtg2
 &end
END
if ( $status != 0 ) then
  echo " ** Error in grdmath.  FATAL ERROR, EXITING SCRIPT ** "
endif
if ($samename == 1) then
  /bin/mv $filecout $filefinalout
endif

DONE:
exit
ERROR:
echo ERROR in grdmath.com
exit 1
```