Instructions for using NEMO on SISU

Yongmei Gong
12 April 2018

The following instruction consist text instructions and a number of well commented bash files on how to launch basic nemo simulations. They are created for the environment of the super computer Sisu in CSC-IT center for science.

Contents

1 Build and use NEMO on Sisu

1.1 Get the code

1. Sign up for the wiki (http://forge.ipsl.jussieu.fr/nemo/wiki/Users) by sending an email containing your username (5 characters minimum length) to nemo@forge.ipsl.jussieu.fr;
2. Get the conformation email then reset your password.
3. Now download the source code from the distribution

```
#!/bin/bash

set -ex

# 2018-02-20, juha.lento@csc.fi'
# modified 2018-03-08, yongmei.gong@helsinki.fi

# NEMO is "research code", which for NEMO means that:
# - "install" in the NEMO documentation actually is what
    is usually referred as "build"

# The following build instructions are based on:
# - https://forge.ipsl.jussieu.fr/nemo/wiki/Users/
    ModelInstall#ExtracttheNEMOcode
# - https://forge.ipsl.jussieu.fr/nemo/wiki/Users/
    ModelInterfacing/InputsOutputs#
    ExtractingandinstallingXIOS

#Your user name used for registering to nemo user wiki
USER="YOUR_USER_NAME"

# Load system I/O libraries (that is how we use an
    existing software on Sisu)
# now use xios2.0 by default

module load cray-hdf5-parallel cray-netcdf-hdf5parallel
    xios/


# Create the main NEMO directory (a folder) where you want
     to store the source code
# Here I put in in the user application directory in Sisu,
     $USERAPPL, where you can build (install) your own
    software

mkdir -p $USERAPPL/nemo_test
cd $USERAPPL/nemo_test
```

```
30
# Checkout (download) the source code
# Type your password on the screen and type yes when asked
    .

svn --username $USER co http://forge.ipsl.jussieu.fr/nemo/
    svn/branches/2015/nemo_v3_6_STABLE/NEMOGCM
```

1.2 Declare the compilers

All compiler options in NEMO are controlled using files in NEMOGCM/ARCH/arch. Now we create a file to declare the compilers we use to build nemo accroding to what we have in Sisu.

```
#!/bin/bash

set -ex

5 # NEMO 3.6 STABLE + XIOS-2 build instructions for sisu.csc
    .fi
#
# 2018-02-20, juha.lento@csc.fi
# modified 2018-03-08, yongmei.gong@helsinki.fi

10 # NEMO is "research code", which for NEMO means that:
# - "install" in the NEMO documentation actually is what
    is usually referred as "build"

# The following build instructions are based on:
# - https://forge.ipsl.jussieu.fr/nemo/wiki/Users/
    ModelInstall#ExtracttheNEMOcode
15 # - https://forge.ipsl.jussieu.fr/nemo/wiki/Users/
    ModelInterfacing/InputsOutputs#
    ExtractingandinstallingXIOS


# NEMO build
# All compiler options in NEMO are controlled using files
    in NEMOGCM/ARCH/arch-'my_arch'.fcm where 'my_arch' is
    the name of the computing architecture.
20 # Now we create a file to declare the compilers we use to
    build nemo accroding to what we have in Sisu

cd $USERAPPL/nemo_test/NEMOGCM
cat > ARCH/arch-XC40-SISU.fcm <<EOF
%NCDF_HOME           $NETCDF_DIR
25 %HDF5_HOME           $HDF5_DIR
```

```
%XIOS_HOME           $(pkg-config --variable=CRAY_prefix
    xios)
%NCDF_INC            -I%NCDF_HOME/include -I%HDF5_HOME/
    include
%NCDF_LIB            -L%HDF5_HOME/lib -L%NCDF_HOME/lib -
    lnetcdff -lnetcdf -lhdf5_hl -lhdf5 -lz
%XIOS_INC            -I%XIOS_HOME/inc
%XIOS_LIB            -L%XIOS_HOME/lib -lxios
%CPP                 cpp
%FC                  ftn
%FCFLAGS             -emf -s real64 -s integer32  -O2 -
    hflex_mp=intolerant -Rb
%FFLAGS              -emf -s real64 -s integer32  -O0 -
    hflex_mp=strict -Rb
%LD                  ftn
%FPPFLAGS            -P -E -traditional-cpp
%LDFLAGS             -hbyteswapio
%AR                  ar
%ARFLAGS             -r
%MK                  gmake
%USER_INC            %XIOS_INC
%USER_LIB            %XIOS_LIB
%CC                  cc
%CFLAGS              -O0
EOF
```

1.3 Build NEMO for experiment - GYRE_XIOS

Now we build an executable for the experiment GYRE_XIOS.
We use the up-to-date version of XIOS - xios2.0.990 instead the oldder xios1.0. This requires declaring the active cpp keys in the cpp_*_.fcm files.

```
#!/bin/bash

set -ex

# NEMO 3.6 STABLE + XIOS-2 build instructions for sisu.csc
    .fi
#
# 2018-02-20, juha.lento@csc.fi
# modified 2018-03-08, yongmei.gong@helsinki.fi

# NEMO is "research code", which for NEMO means that:
# - "install" in the NEMO documentation actually is what
    is usually referred as "build"

# The following build instructions are based on:
```

```
# - https://forge.ipsl.jussieu.fr/nemo/wiki/Users/ModelInstall#ExtracttheNEMOcode
# - https://forge.ipsl.jussieu.fr/nemo/wiki/Users/ModelInterfacing/InputsOutputs#ExtractingandinstallingXIOS

# Load system I/O libraries
# might change to xios2.0

module load cray-hdf5-parallel cray-netcdf-hdf5parallel xios/2.0.990


# Declare your NEMO code directory

NEMOBUILD="$USERAPPL/nemo_test3"

# NEMO build

# Declare your configuration for the simulation, e.g. the GYRE_XIOS experiment
cd $NEMOBUILD/NEMOGCM/CONFIG

#you need to add new keys in .fcm file
sed -i 's/$/ key_nosignedzero/' GYRE_XIOS/cpp_GYRE_XIOS.fcm

# Here you compile a executable for the experiment GYRE_XIOS in either $TMPDIR or $WRKDIR/DONOTREMOVE
./makenemo -t $TMPDIR -m XC40-SISU -r GYRE_XIOS -n MY_GYRE_XIOS

# NEMO test
#
# For a quick test, only!

# cd MY_GYRE_XIOS/EXP00
# cp $TMPDIR/MY_GYRE_XIOS/BLD/bin/nemo.exe .
# aprun -n 4 nemo.exe


# For actual experiments:
# Copy the experiments in $WRKDIR/DONOTREMOVE
cp GYRE_XIOS/ $WRKDIR/DONOTREMOVE/MY_GYRE_XIOS/
cd $WRKDIR/DONOTREMOVE/MY_GYRE_XIOS/EXP00

# Copy the executable to the EXP00 directory
cp $TMPDIR/MY_GYRE_XIOS/BLD/bin/nemo.exe .
```

```
# Creat a script for Using SLURM commands to execute batch
    jobs
 in Sisu queue
# More about the SLURM commands can be found in
# - https://research.csc.fi/sisu-using-slurm-commands-to-
   execute-batch-jobs

cat > batch_job.sh <<EOF
#!/bin/bash -l
#SBATCH -t 00:29:00
#SBATCH -J gyre_xios
#SBATCH -p test
#SBATCH -o gyre_xios.%j
#SBATCH -e gyre_xios_err.%j
#SBATCH -N 4

aprun -n 4 nemo.exe
EOF

# Then submit the job in the queue
sbatch batch_job.sh
```

1.4 Build NEMO for experiment - ORCA2_LIM3

Now we build an executable for the experiment ORCA2_LIM3.
ORCA is the generic name given to global ocean configurations. Its specificity lies on the horizontal curvilinear mesh used to overcome the North Pole singularity found for geographical meshes. LIM (Louvain-la-Neuve sea-ice model, multi-category model LIM3 is used) is a thermodynamic-dynamic sea ice model specifically designed for climate studies.
Similarly this requires declaring the active cpp keys in the cpp_*_.fcm files. And this time the experiment needs input data.

```
#!/bin/bash

set -ex

# NEMO 3.6 STABLE + XIOS-2 build instructions for sisu.csc
   .fi
#
# 2018-02-20, juha.lento@csc.fi
# modified 2018-03-08, yongmei.gong@helsinki.fi

# NEMO is "research code", which for NEMO means that:
# - "install" in the NEMO documentation actually is what
   is usually referred as "build"
```

```
# The following build instructions are based on:
# - https://forge.ipsl.jussieu.fr/nemo/wiki/Users/ModelInstall#ExtracttheNEMOcode
# - https://forge.ipsl.jussieu.fr/nemo/wiki/Users/ModelInterfacing/InputsOutputs#ExtractingandinstallingXIOS

# Load system I/O libraries
# might change to xios2.0

module load cray-hdf5-parallel cray-netcdf-hdf5parallel xios/2.0.990


# Declare your NEMO code directory

NEMOBUILD="$USERAPPL/nemo_test3"

# NEMO build

# Declare your configuration for the simulation, e.g. the ORCA2_LM3 experiment
cd $NEMOBUILD/NEMOGCM/CONFIG
#you need to add new keys in .fcm file
sed -i 's/$/ key_nosignedzero key_xios2/' ORCA2_LIM3/cpp_ORCA2_LIM3.fcm

# Here you compile a executable for the experiment ORCA2_LIM3 in $TMPDIR

./makenemo -t $TMPDIR -m XC40-SISU -r ORCA2_LIM3 -n MY_ORCA2_LIM3


# NEMO test
#
cd ORCA2_LIM3/EXP00
# If you use XIOS2.0 you need to copy all the .xml file from GYRE_XIOS
cp $NEMOBUILD/NEMOGCM/CONFIG/GYRE_XIOS/EXP00/*xml .


# For actual experiments:
# Copy the experiments in $WRKDIR/DONOTREMOVE
cd ../..
cp ORCA2_LIM3/ $WRKDIR/DONOTREMOVE/MY_ORCA2_LIM3/
cd $WRKDIR/DONOTREMOVE/MY_ORCA2_LIM3/
cp $TMPDIR/MY_ORCA2_LIM3/BLD/bin/nemo.exe .
```

```
# now you need to download input data ORCA2_LIM_nemo_v3.6.
    tar from http://forge.ipsl.jussieu.fr/nemo/wiki/Users/
    ReferenceConfigurations/ORCA2_LIM3_PISCES
#
# Then extract them in EXP00
tar xvf ORCA2_LIM_nemo_v3.6.tar
gzip -d *gz

# Creat a script for Using SLURM commands to execute batch
     jobs
 in Sisu queue
# More about the SLURM commands can be found in
# - https://research.csc.fi/sisu-using-slurm-commands-to-
    execute-batch-jobs
cat > batch_job.sh <<EOF
#!/bin/bash -l
#SBATCH -t 00:29:00
#SBATCH -J ORCA2_LIM3
#SBATCH -p test
#SBATCH -o ORCA2_LIM3.%j
#SBATCH -e ORCA2_LIM3_err.%j
#SBATCH -N 4

aprun -n 4 nemo.exe
EOF

# Then submit the job in the queue
sbatch batch_job.sh
```

1.5 Build NEMO for experiment - ORCA1_LIM3

The difference between ORCA1 and ORCA2 is that the spacial resolution of the former is 1 degree and the latter is 2 degree. So ORCA1 has higher resolution and also needs more input data.
You will need files and data in two folders: ORCA1_cfg and ORCA1_data to create ORCA1 experiment configuration

```
#!/bin/bash

set -ex

# NEMO 3.6 STABLE + XIOS-2 build instructions for sisu.csc
    .fi
#
# 2018-02-20, juha.lento@csc.fi
# modified 2018-03-06, yongmei.gong@helsinki.fi
```

```
# NEMO is "research code", which for NEMO means that:
# - "install" in the NEMO documentation actually is what is usually referred as "build"

# The following build instructions are based on:
# - https://forge.ipsl.jussieu.fr/nemo/wiki/Users/ModelInstall#ExtracttheNEMOcode
# - https://forge.ipsl.jussieu.fr/nemo/wiki/Users/ModelInterfacing/InputsOutputs#ExtractingandinstallingXIOS

# Load system I/O libraries
# might change to xios2.0

module load cray-hdf5-parallel cray-netcdf-hdf5parallel xios/2.0.990


# Declare your NEMO code directory

NEMOBUILD="$USERAPPL/nemo_test3"

# NEMO build

# There is no ORCA1 experiment, which means in the standard distribution therefore we need to creat it ourselves

cd $NEMOBUILD/NEMOGCM/CONFIG
mkdir ORCA1
cd ORCA1

# put the cpp file here
cp /homeappl/home/ygong/appl_sisu/NEMO/NEMO_local/ORCA1_cfg/cpp_ORCA1_LIM3.fcm .

# Creat a experiment dir EXP00 and put all the xml and namelist files and data there
# Make sure that you have all the file_* and field_* xml files listed in context_nemo.xml (you don't need * pisces.xml files if you don't want to couple with biochemistry).

# you can put the line 'nn_msh      =    0      !  create (=1) a mesh file or not (=0)' under &namdom in the file namelist_cfg.
# This prevents nemo to creat the mesh files, which will cause trouble when you re-launch the experiment in the
```

```
    same directory as they cannot be over-written

mkdir EXP00
cp /homeappl/home/ygong/appl_sisu/NEMO/NEMO_local/
    ORCA1_cfg/*xml .
cp /homeappl/home/ygong/appl_sisu/NEMO/NEMO_local/
    ORCA1_cfg/namelist* .

# Here you compile a executable for the experiment ORCA1
    in $TMPDIR
cd ..
./makenemo -t $TMPDIR -m XC40-SISU -r ORCA1 -n MY_ORCA1



# NEMO test
# now the experiment gets bigger so we do it in $WRKDIR
cd $WRKDIR/DONOTREMOVE/
mkdir MY_ORCA1
cd ORCA1
cp $NEMOBUILD/NEMOGCM/CONFIG/ORCA1/EXP00 .
cp $TMPDIR/MY_ORCA2_LIM3/BLD/bin/nemo.exe .

# Data includes all the data describing the boundray and
    initial conditions and the climatology in NetCDF format
    . And a samll *.dat file describing the humidity
cp /homeappl/home/ygong/appl_sisu/NEMO/NEMO_local/
    ORCA1_data/* .

# Creat a script for Using SLURM commands to execute batch
     jobs
 in Sisu queue
# More about the SLURM commands can be found in
# - https://research.csc.fi/sisu-using-slurm-commands-to-
    execute-batch-jobs

cat > batch_job.sh <<EOF
#!/bin/bash -l
#SBATCH -t 12:00:00
#SBATCH -J ORCA1
#SBATCH -p small
#SBATCH -o ORCA1.%j
#SBATCH -e ORCA1_err.%j
#SBATCH -N 4

aprun -n 24 nemo.exe
EOF

# submit the job
```

```
sbatch batch_job.sh
```

1.6 Use nemo for long transient runs

Note below is just a template file. To be able to run it you need additional files which you can get by:
>>git clone https://version.helsinki.fi/hydro_geophysics/nemo_scripts.git
However you need to get the permission to access.
Then you need to ask data from Petteri.

```
#!//bin/bash -l

#SBATCH -t 02:00:00
#SBATCH -J N101.sh
#SBATCH -p small
#SBATCH -o output_%j.txt
#SBATCH -e errors_%j.txt
#SBATCH -N 4

## the last line is to

set -ue

# librunscript defines some helper functions
source ./librunscript.sh

#
    ================================================================================

# *** BEGIN User configuration
#
    ================================================================================


#
    --------------------------------------------------------------------------------

# *** Preload modules in sisu
#
    --------------------------------------------------------------------------------


pre_load_modules_cmd="module load cray-hdf5-parallel cray-
    netcdf-hdf5parallel xios/2.0.990"
${pre_load_modules_cmd}
```

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

# *** General configuration
#
   ------------------------------------------------------------------------


# Component configuration
# (for syntax of the $config variable, see librunscript.sh
    )
config="nemo lim3 xios:detached"

# Experiment name (exactly 4 letters!)
exp_name=N103
nem_forcing_set=DFS5.2
nem_forcing_dir=/wrk/puotila/DONOTREMOVE/${nem_forcing_set
    }

# Simulation start and end date. Use any (reasonable)
    syntax you want.
run_start_date="1990-01-01"
run_end_date="${run_start_date} + 4 months" #

# Set $force_run_from_scratch to 'true' if you want to
    force this run to start
# from scratch, possibly ignoring any restart files
    present in the run
# directory. Leave set to 'false' otherwise.
# NOTE: If set to 'true' the run directory $run_dir is
    cleaned!
force_run_from_scratch=false #true

# Resolution
nem_grid=ORCA1L75

# Restart frequency. Use any (reasonable) number and time
    unit you want.
# For runs without restart, leave this variable empty
# Normally you need it if you want to resubmit jobs
    through the function Finalize
rst_freq="1 month"

# Number of restart legs to be run in one job
run_num_legs=2

# Directories
```

```
# Tells where to find the restart files if the first round of simulation is restarted from some other files.
# And also tells config file to find the run script to resubmit the job etc
start_dir=${PWD}

# Tells where to find namelist and xml etc. files
ctrl_file_dir=${USERAPPL}/NEMO/NEMO_local/ORCA1_cfg

# Tells where to find nemo executable
nem_src_dir=${WRKDIR}/DONOTREMOVE/NEMOEXP

# Architecture
#build_arch=my_ecconf   #?? it has been used anywhere ??

# This file is in the run dir and used to store information about restarts
# If the first run leg is not restarting from anything it will be created in the run dir
nem_info_file="nem.info"   #?? should we change the name of the variable to avoid confusion ??

#
   ------------------------------------------------------------------------

# *** Read platform dependent configuration
#
   ------------------------------------------------------------------------

source ./nemconf.cfg  # read function configure from ecconf.cfg ?? probably need another name to avoid confusio??

configure

#
   ------------------------------------------------------------------------

# *** Time step settings
#
   ------------------------------------------------------------------------

case "${nem_grid}" in

    ORCA1L*)   nem_time_step_sec=2700; lim_time_step_sec=2700 ;;
    ORCA025L*) nem_time_step_sec=900 ; lim_time_step_sec=900  ;;
```

```
    *)  error "Can't set time steps for unknown horizontal
         grid: ${nem_grid}"
        ;;
esac

#
   ------------------------------------------------------------------------

# *** NEMO/LIM configuration
#
   ------------------------------------------------------------------------


# This is only needed if the experiment is started from an
     existing set of NEMO
# restart files
nem_restart_file_path=${start_dir}/nemo-rst #!!! need to
    be changed

nem_restart_offset=0 #-607360

nem_res_hor=$(echo ${nem_grid} | sed 's:ORCA\([0-9]\+\)L
    [0-9]\+:\1:')

# Pick correct NEMO configuration, which is one of:
#   NEMO standalone, NEMO+PISCES-standaone, PISCES-offline
                              nem_config_name=${nem_grid}
                                  _LIM3_standalone
has_config pisces         && nem_config_name=${nem_grid}
    _LIM3_PISCES_standalone
has_config pisces:offline && nem_config_name=${nem_grid}
    _OFF_PISCES

nem_exe_file=${nem_src_dir}/${nem_config_name}/BLD/bin/
    nemo.exe
nem_numproc=72 #48

#
   ------------------------------------------------------------------------

# *** XIOS configuration
#
   ------------------------------------------------------------------------


# Now we preload xios
xio_exe_file=$(which xios_server.exe)
```

```
xio_numproc=2 #0  #??? don't really know, should change 0
    to some integer if xios is not detached ???

#
    ------------------------------------------------------------------------

# *** atmospheric model configuration
#
    ------------------------------------------------------------------------


#ifs_exe_file=amt_test.exe
#ifs_numproc=3
#ifs_key="-v ecmwf -e"

#
    ========================================================================

# *** END of User configuration
#
    ========================================================================


#
    ========================================================================

# *** This is where the code begins ...
#
    ========================================================================


#
    ------------------------------------------------------------------------

# *** Create the run dir if necessary and go there
#     Everything is done from here.
#
    ------------------------------------------------------------------------

if [ ! -d ${run_dir} ]
then
    mkdir -p ${run_dir}
fi
cd ${run_dir}

#
    ------------------------------------------------------------------------

# *** Determine the time span of this run and whether it's
```

```
    a restart leg
#   ------------------------------------------------------------------------


# Regularise the format of the start and end date of the simulation
run_start_date=$(absolute_date_noleap "${run_start_date}")
run_end_date=$(absolute_date_noleap "${run_end_date}")

# Loop over the number of legs
for (( ; run_num_legs>0 ; run_num_legs-- ))
do

    # Check for restart information file and set the current leg start date
    #   Ignore restart information file if force_run_from_scratch is true
    if ${force_run_from_scratch} || ! [ -r ${nem_info_file} ]
    then
        leg_is_restart=false
        leg_start_date=${run_start_date}
        leg_number=1
    else
        leg_is_restart=true
        . ./${nem_info_file}
        leg_start_date=${leg_end_date}
        leg_number=$((leg_number+1))
    fi

    # Compute the end date of the current leg
    if [ -n "${rst_freq}" ]
    then
        leg_end_date=$(absolute_date_noleap "${leg_start_date} + ${rst_freq}")
    else
        leg_end_date=${run_end_date}
    fi

    if [ $(date -d "${leg_end_date}" +%s) -gt $(date -d "${run_end_date}" +%s) ]
    then
        leg_end_date=${run_end_date}
    fi

    # Some time variables needed later
    leg_length_sec=$(( $(date -d "${leg_end_date}" +%s) - $(date -d "${leg_start_date}" +%s) ))
```

```
    leg_start_sec=$(( $(date -d "${leg_start_date}" +%s) -
        $(date -d "${run_start_date}" +%s) ))
    leg_end_sec=$(( $(date -d "${leg_end_date}" +%s) - $(
        date -d "${run_start_date}" +%s) ))
    leg_start_date_yyyymmdd=$(date -u -d "${leg_start_date
        }" +%Y%m%d)
    leg_start_date_yyyy=$(date -u -d "${leg_start_date}"
        +%Y)
    leg_end_date_yyyy=$(date -u -d "${leg_end_date}" +%Y)

    # Correct for leap days because NEMO standalone uses
        no-leap calendar
    leg_length_sec=$(( leg_length_sec - $(leap_days "${
        leg_start_date}" "${leg_end_date}")*24*3600 ))
    leg_start_sec=$(( leg_start_sec - $(leap_days "${
        run_start_date}" "${leg_start_date}")*24*3600 ))
    leg_end_sec=$(( leg_end_sec - $(leap_days "${
        run_start_date}" "${leg_end_date}")*24*3600 ))

    # Check whether there's actually time left to simulate
        - exit otherwise
    if [ ${leg_length_sec} -le 0 ]
    then
        info "Leg start date equal to or after end of
            simulation."
        info "Nothing left to do. Exiting."
        exit 0
    fi

    #
        ------------------------------------------------------------

    # *** Prepare the run directory for a run from scratch
    #
        ------------------------------------------------------------

    if ! $leg_is_restart
    then
        #
            ------------------------------------------------------

        # *** Check if run dir is empty. If not, and if we
            are allowed to do so
        #     by ${force_run_from_scratch}, remove
            everything
        #
            ------------------------------------------------------

        if $(ls * >& /dev/null)
```

```
        then
            if ${force_run_from_scratch}
            then
                rm -fr ${run_dir}/*
            else
                error "Run directory not empty and \
                    $force_run_from_scratch not set."
            fi
        fi

        #
           ____________________________________________________________________

        # *** Copy executables of model components
        # *** Additionally, create symlinks to the
            original place for reference
        #
           ____________________________________________________________________

        cp    ${nem_exe_file} .
        ln -s ${nem_exe_file} $(basename ${nem_exe_file}).
            lnk

        cp    ${xio_exe_file} .
        ln -s ${xio_exe_file} $(basename ${xio_exe_file}).
            lnk

        #
           ____________________________________________________________________

        # *** Files needed for NEMO (linked)
        #
           ____________________________________________________________________



        # Link initialisation files for matching ORCA grid
             ?? put these files somewhere? rundir?
        for f in \
            bathy_meter.nc coordinates.nc \
            ahmcoef.nc \
            K1rowdrg.nc M2rowdrg.nc mask_itf.nc \
            decay_scale_bot.nc decay_scale_cri.nc \
            mixing_power_bot.nc mixing_power_cri.nc
                mixing_power_pyc.nc \
            runoff_depth.nc
        do
            [ -f ${ini_data_dir}/nemo/initial/${nem_grid}/
                $f ] && ln -s ${ini_data_dir}/nemo/initial/
                ${nem_grid}/$f
```

```
        done

        # Link geothermal heating file (independent of
            grid) and matching weight file
        ln -s ${ini_data_dir}/nemo/initial/Goutorbe_ghflux
            .nc
        ln -s ${ini_data_dir}/nemo/initial/
            weights_Goutorbe1_2_orca${nem_res_hor}_bilinear
            .nc

        # Link either restart files or climatology files
            for the initial state
        if $(has_config nemo:start_from_restart)
        then
            # When linking restart files, we accept three
                options:
            # (1) Merged files for ocean and ice, i.e.
            #     restart_oce.nc and restart_ice.nc
            # (2) One-file-per-MPI-rank, i.e.
            #     restart_oce_????.nc and restart_ice_
                ????.nc
            #     No check is done whether the number of
                restart files agrees
            #     with the number of MPI ranks for NEMO!
            # (3) One-file-per-MPI-rank with a prefix, i.e
                .
            #     <exp_name>_<time_step>_restart_oce_????.
                nc (similar for the ice)
            #     The prefix is ignored.
            # The code assumes that one of the options can
                be applied! If more
            # options are applicable, the first is chosen.
                If none of the
            # options apply, NEMO will crash with missing
                restart file.
            if   ls -U ${nem_restart_file_path}/restart_[
                oi]ce.nc > /dev/null 2>&1
            then
                 ln -s ${nem_restart_file_path}/restart_[
                     oi]ce.nc ./

            elif ls -U ${nem_restart_file_path}/restart_[
                oi]ce_????.nc > /dev/null 2>&1
            then
                 ln -s ${nem_restart_file_path}/restart_[
                     oi]ce_????.nc ./

            else
                for f in ${nem_restart_file_path}/????_
```

```
                ????????_restart_[oi]ce_????.nc
            do
                ln -s $f $(echo $f | sed 's/.*_\(
                    restart_[oi]ce_....\.nc\)/\1/')
            done
        fi
    else

        # Temperature and salinity files for
            initialisation
        ln -s ${ini_data_dir}/nemo/climatology/
            absolute_salinity_WOA13_decav_Reg1L75_clim.
            nc
        ln -s ${ini_data_dir}/nemo/climatology/
            conservative_temperature_WOA13_decav_Reg1L75_clim
            .nc
        ln -s ${ini_data_dir}/nemo/climatology/
            weights_WOA13d1_2_orca${nem_res_hor}
            _bilinear.nc

        # Grid dependent runoff files
        case ${nem_grid} in
        ORCA1*)   ln -s ${ini_data_dir}/nemo/
            climatology/runoff-
            icb_DaiTrenberth_Depoorter_ORCA1_JD.nc ;;
        ORCA025*) ln -s ${ini_data_dir}/nemo/
            climatology/ORCA_R025_runoff_v1.1.nc ;;
        esac
    fi

    # Write fake file for previous fresh water budget
        adjustment (nn_fwb==2 in namelist)
    echo "                                   0
        0.0000000000000000E+00  0.0000000000000000E+00"
         > EMPave_old.dat


    #
        ____________________________________________________

    # *** Link atmospheric forcing files for this leg
    #
        ____________________________________________________

case ${nem_forcing_set} in
    DFS5.2)
  for v in u10 v10 t2 q2 precip snow radlw radsw; do
    for i in $(eval echo {$leg_start_date_yyyy..
        $leg_end_date_yyyy}); do
```

```
      ln -fs ${nem_forcing_dir}/drowned_${v}_DFS5.2_y${i}.
          nc ./${v}_y${i}.nc
        done
    done
                # Link DFS52 weight files for
                    corresponding grid
                # Weight files for forcing
    ln -sf ${nem_forcing_dir}/weights_${nem_forcing_set}
        _orca${nem_res_hor}_bilinear.nc .
    ln -sf ${nem_forcing_dir}/weights_${nem_forcing_set}
        _orca${nem_res_hor}_bicubic.nc .
    ;;
      *)
                # Link NEMO CoreII forcing files (only set
                     supported out-of-the-box)
    for v in u_10 v_10 t_10 q_10 ncar_precip ncar_rad
    do
        f="${ini_data_dir}/nemo/forcing/CoreII/${v}.15
            JUNE2009_fill.nc"
        [ -f "$f" ] && ln -s $f
    done
                # Link CoreII weight files for
                    corresponding grid
    ln -s ${ini_data_dir}/nemo/forcing/CoreII/
        weights_coreII_2_orca${nem_res_hor}_bilinear.nc
    ln -s ${ini_data_dir}/nemo/forcing/CoreII/
        weights_coreII_2_orca${nem_res_hor}_bicubic.nc
    ;;
  esac

        # XIOS files
        . ${ctrl_file_dir}/iodef.xml.sh > iodef.xml
  #ln -s ${ctrl_file_dir}/iodef.xml
        ln -s ${ctrl_file_dir}/context_nemo.xml
        ln -s ${ctrl_file_dir}/domain_def_nemo.xml
        ln -s ${ctrl_file_dir}/field_def_nemo-lim.xml
        ln -s ${ctrl_file_dir}/field_def_nemo-opa.xml
        ln -s ${ctrl_file_dir}/field_def_nemo-pisces.xml
        ln -s ${ctrl_file_dir}/file_def_nemo-lim3.xml
            file_def_nemo-lim.xml #
  #ln -s ${ctrl_file_dir}/file_def_nemo-lim.xml
        ln -s ${ctrl_file_dir}/file_def_nemo-opa.xml
        ln -s ${ctrl_file_dir}/file_def_nemo-pisces.xml

        #
            ------------------------------------------------------------

        # *** Files needed for TOP/PISCES (linked)
        #
```

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

        if $(has_config pisces)
        then
            ln -fs ${ini_data_dir}/pisces/
                dust_INCA_ORCA_R1.nc
            ln -fs ${ini_data_dir}/pisces/
                ndeposition_Duce_ORCA_R1.nc
            ln -fs ${ini_data_dir}/pisces/
                pmarge_etopo_ORCA_R1.nc
            ln -fs ${ini_data_dir}/pisces/
                river_global_news_ORCA_R1.nc
            ln -fs ${ini_data_dir}/pisces/
                Solubility_T62_Mahowald_ORCA_R1.nc

            ln -fs ${ini_data_dir}/pisces/
                par_fraction_gewex_clim90s00s_ORCA_R1.nc
            ln -fs ${ini_data_dir}/pisces/
                DIC_GLODAP_annual_ORCA_R1.nc
            ln -fs ${ini_data_dir}/pisces/
                Alkalini_GLODAP_annual_ORCA_R1.nc
            ln -fs ${ini_data_dir}/pisces/
                O2_WOA2009_monthly_ORCA_R1.nc
            ln -fs ${ini_data_dir}/pisces/
                PO4_WOA2009_monthly_ORCA_R1.nc
            ln -fs ${ini_data_dir}/pisces/
                Si_WOA2009_monthly_ORCA_R1.nc
            ln -fs ${ini_data_dir}/pisces/
                DOC_PISCES_monthly_ORCA_R1.nc
            ln -fs ${ini_data_dir}/pisces/
                Fer_PISCES_monthly_ORCA_R1.nc
            ln -fs ${ini_data_dir}/pisces/
                NO3_WOA2009_monthly_ORCA_R1.nc
        fi

    else # i.e. $leg_is_restart == true

        #-------------------------------------------------------------------

        # *** Remove all leftover output files from
            previous legs
        #-------------------------------------------------------------------


        # NEMO output files
        rm -f ${exp_name}_??_????????_????????_{grid_U,
            grid_V,grid_W,grid_T,icemod,SBC,scalar,
```

```
            SBC_scalar,diad_T}.nc

    fi # ! $leg_is_restart

    #
       ____________________________________________________________________

    # *** Create some control files
    #
       ____________________________________________________________________


    # Remove land grid-points
    if $(has_config nemo:elpin)
    then
        jpns=($(${nemctrl_scr_dir}/util/ELPiN/ELPiNv2.cmd
            ${nem_numproc}))
        info "nemo domain decompostion from ELpIN: ${jpns[
            @]}"
        nem_numproc=${jpns[0]}
        nem_jpni=${jpns[1]}
        nem_jpnj=${jpns[2]}
    else
        info "nemo original domain decomposition (not
            using ELPiN)"
    fi

    # NEMO and LIM namelists

    . ${ctrl_file_dir}/namelist.nemo.ref.sh
                               > namelist_ref
    . ${ctrl_file_dir}/namelist.nemo-${nem_grid}-standalone
       .cfg.sh > namelist_cfg
    . ${ctrl_file_dir}/namelist.lim3.ref.sh
                               > namelist_ice_ref
    . ${ctrl_file_dir}/namelist.lim3-${nem_grid}.cfg.sh
                  > namelist_ice_cfg

    # NEMO/TOP+PISCES namelists # !!! need to change this
        in the future
    has_config pisces && . ${ctrl_file_dir}/namelist.nemo.
        top.ref.sh    > namelist_top_ref
    has_config pisces && . ${ctrl_file_dir}/namelist.nemo.
        top.cfg.sh    > namelist_top_cfg
    has_config pisces && . ${ctrl_file_dir}/namelist.nemo.
        pisces.ref.sh > namelist_pisces_ref
    has_config pisces && . ${ctrl_file_dir}/namelist.nemo.
        pisces.cfg.sh > namelist_pisces_cfg
```

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

    # *** Link the appropriate NEMO restart files of the
       previous leg
    #
       ------------------------------------------------------------------

    if $leg_is_restart
    then
        ns=$(printf %08d $(( leg_start_sec /
            nem_time_step_sec - nem_restart_offset )))
        for (( n=0 ; n<nem_numproc ; n++ ))
        do
            np=$(printf %04d ${n})
            ln -fs ${exp_name}_${ns}_restart_oce_${np}.nc
                restart_oce_${np}.nc
            ln -fs ${exp_name}_${ns}_restart_ice_${np}.nc
                restart_ice_${np}.nc
            has_config pisces && \
            ln -fs ${exp_name}_${ns}_restart_trc_${np}.nc
                restart_trc_${np}.nc
        done

        # Make sure there are no global restart files
        # If links are found, they will be removed. We are
            cautious and do
        # _not_ remove real files! However, if real global
            restart files are
        # present, NEMO/LIM will stop because time stamps
           will not match.
        [ -h restart_oce.nc ] && rm restart_oce.nc
        [ -h restart_ice.nc ] && rm restart_ice.nc
    fi

    ### !!!! okay this needs to be changed!!!!
    #
       ------------------------------------------------------------------

    # *** Start the run
    #
       ------------------------------------------------------------------



    # Use the launch function from the platform
       configuration file

    t1=$(date +%s)
```

```
    launch \
        ${xio_numproc} ${xio_exe_file} -- \
        ${nem_numproc} ${nem_exe_file} #-- \
#${ifs_numproc} ${ifs_exe_file} ${ifs_key} $exp_name
  #aprun -n ${nem_numproc} ${nem_exe_file}:
  t2=$(date +%s)

  tr=$(date -d "0 -$t1 sec + $t2 sec" +%T)

  #
     ------------------------------------------------------------------

  # *** Check for signs of success
  #     Note the tests provide no guarantee that things
     went fine! They are
  #     just based on the IFS and NEMO log files. More
     tests (e.g. checking
  #     restart files) could be implemented.
  #
     ------------------------------------------------------------------

  # Check for NEMO success
  if [ -f ocean.output ]
  then
      if [ "$(awk '/New day/{d=$10}END{print d}' ocean.
         output)" == "$(date -u -d "$(
         absolute_date_noleap "${leg_end_date} - 1 day")
         " +%Y/%m/%d)" ]
      then
          info "Leg successfully completed according to
             NEMO log file 'ocean.output'."
      else
          error "Leg not completed according to NEMO log
               file 'ocean.output'."
      fi

  else
      error "NEMO log file 'ocean.output' not found
         after run."
  fi

  #
     ------------------------------------------------------------------

  # *** Move NEMO output files to archive directory
  #
     ------------------------------------------------------------------

  outdir="output/nemo/$(printf %03d $((leg_number)))"
```

```
    mkdir -p ${outdir}

    for v in grid_U grid_V grid_W grid_T icemod SBC scalar
         SBC_scalar diad_T ptrc_T
    do
        for f in ${exp_name}_*_????????_????????_${v}.nc
        do
            test -f $f && mv $f $outdir/
        done
    done

    #
       ------------------------------------------------------------------

    # *** Move NEMO restart files to archive directory
    #
       ------------------------------------------------------------------


    outdir="restart/nemo/$(printf %03d $((leg_number)))"
    mkdir -p ${outdir}

    ns=$(printf %08d $(( leg_start_sec / nem_time_step_sec
         - nem_restart_offset )))

    for f in oce ice trc
    do
  for i in ${exp_name}_${ns}_restart_${f}_????.nc
  do
      test -f $i && mv $i $outdir/
  done
    done


    #
       ------------------------------------------------------------------

    # *** Move log files to archive directory
    #
       ------------------------------------------------------------------

    outdir="log/$(printf %03d $((leg_number)))"
    mkdir -p ${outdir}

    for f in \
        ocean.output time.step solver.stat
    do
        test -f ${f} && mv ${f} ${outdir}
    done
```

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

    # *** Write the restart control file
    #
       ------------------------------------------------------------------


    # Now nem_info_file is created in the run dir
    # Compute CPMIP performance
    sypd="$(cpmip_sypd $leg_length_sec $(($t2 - $t1)))"
    chpsy="$(cpmip_chpsy  $leg_length_sec $(($t2 - $t1)) $
        (($nem_numproc + $xio_numproc)))"

    echo "#"                                                  |
         tee -a ${nem_info_file}
    echo "# Finished leg at `date '+%F %T'` after ${tr} (
        hh:mm:ss)" \
                                                             |
                                                               tee
                                                               -
                                                               a
                                                               $
                                                               {
                                                               nem_info_file
                                                               }
    echo "# CPMIP performance: $sypd SYPD   $chpsy CHPSY"|
         tee -a ${nem_info_file}
    echo "leg_number=${leg_number}"                           |
         tee -a ${nem_info_file}
    echo "leg_start_date=\"${leg_start_date}\""               |
         tee -a ${nem_info_file}
    echo "leg_end_date=\"${leg_end_date}\""                   |
         tee -a ${nem_info_file}

    # Need to reset force_run_from_scratch in order to
        avoid destroying the next leg
    force_run_from_scratch=false

done # loop over legs

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

```
# *** Platform dependent finalising of the run
#
   ------------------------------------------------------------------------

finalise \
   $SLURM_JOB_NAME #give the name of the current shell
      script

exit 0
```

1.7 Clean nemo build

If something has gone wrong with or has been changed for the build of nemo (nemo.exe)
- Clean a bad configuration
>>./makenemo -C YOUR_CONFIG clean_config
- Uninstalling (Clean up the whole thing)
>>./makenemo -n YOUR_BUILD clean
(e.g. >>./makenemo ./makenemo -t $TMPDIR -m XC40-SISU -n MY_GYRE_XIOS clean)

1.8 Check the run status and outputs

- If run crushes, try to find 'E R R O R' section in ocean.output - Use ncview (now it is only in Petteris directory) to check the results *nc files
>>cd /homeappl/home/puotila/bin
>>export HDF5_DISABLE_VERSION_CHECK=1
>>./ncview /YOUR_OUTPUT_DIR/_5d_00010101_00011230_grid_T.nc &

1.9 Build nemo tools

TBA

2 Build and use barakuda on Taito

It is the best to build barakuda on taito

2.1 Get the code

Get the version from Petteri's git (you need permission)

```
>> git clone https://version.helsinki.fi/pjuotila/barakuda.git
```

2.2 build up basemap

Before you get barakuda running you need to add lots of module packages for python.

```
#!/bin/bash

set -ex
#####################################
# 2018-04-11, yongmei.gong@helsinki.fi
# more details go there
#https://matplotlib.org/basemap/users/installing.html#installation
#####################################


# !!!login to taito-login4
# load modules needed
module load python-env/intelpython3.5 geos/3.6.1 proj4/4.9.3 hdf5-par netcdf4

##############
# set up python
pip install --upgrade pip --user # first upgrade pip
pip list --outdated --format=freeze | xargs -n1 pip install # then upgrade python modules
pip install netCDF4 --user # install netCDF module
pip install pyproj --user # install pyproj
pip install pyshp --user

#############
# install basemap

# get the code
wget https://github.com/matplotlib/basemap/archive/v1.1.0.tar.gz
# untar
tar -zxf v1.1.0.tar.gz
# tell it where to find geos
export GEOS_DIR=/appl/earth/geos/3.6.1
```

```
# install
cd basemap-1.1.0/
python3.5 setup.py install --user
# test
cd examples/
python3.5 test_rotpole.py
```

2.3 build up cdftools

Go to folder cdftools_light.
1. make sure you have these modules. Otherwise load them. If can't make sure you login to taito-login4.
>> module load intel python-env/intelpython3.5 hdf5-par netcdf4
2. link the make file.
>> ln -s make.macro macro/macro.ifort_taito
3. Make sure you use intel and then make
>> gmake

2.4 Use barakuda

1. Create your configuration file.
For example use configs/config_ORCA1_L75_taito.sh
Change NEMO_OUT_STRCT, DIAG_DIR, CONF_INI_DIR accordingly
2. Get data from Petteri and put them in CONF_INI_DIR
3. Tell taito where to find Petteri's NCO
>> export PATH=$PATH:/homeappl/home/puotila/bin/ncks
4. Then launch the code with your own configuration file e.g. ORCA1_L75_YG_taito and your experiment

```
#!//bin/bash -l

#SBATCH -t 08:00:00
#SBATCH -J barakuda
#SBATCH -o barakuda_out.%j
#SBATCH -e barakuda_err.%j
#SBATCH -p longrun
#SBATCH --mem-per-cpu=64300
#SBATCH -N 1

#cd ${WRKDIR}/DONOTREMOVE/barakuda
module load hdf5-par netcdf4 python-env/intelpython3.5
```

```
./barakuda.sh -C ORCA1_L75_YG_taito -R N157

exit 0
```

3 Build and use Opendrift on taito

OpenDrift is a software for modeling the trajectories and fate of objects or substances drifting in the ocean, or even in the atmosphere.
You can get very detailed instructions here on how to build and use it:
https://github.com/opendrift/opendrift/wiki
There are several things to do before building it on Taito.
Note that OpenDrift (as well as some of the requirements) does not support Python3, so that I use python-env/intelpython2.7-2018.2
>>module load geo-env ffmpeg imagemagick
Install miniconda to be able to get the modules in your user local
Get it from here: https://conda.io/miniconda.html
Then install
>>bash Miniconda3-latest-Linux-x86_64.sh
Follow the instruction then when it asks the installing path type:
>>$HOME/.local/bin/miniconda2/
Then follow the instructions again.
When finished check if you are using miniconda in your .local/bin/mminiconda2 or not:
>>which conda
Also check if you are using the python in you miniconda copy
>>which python
Now install the module packages. NOTE that most of the listed modules are installed already in the python2.7 in geo-env. There are only four missing (see below).
>>conda install –yes hdf4 basemap configobj ffmpeg
Then export the path and build.
>>export PATH=$PATH:$OPENDRIFT_FOLDER/opendrift/scripts/
>>python setup.py develop –user
export GDAL_DATA equal to the folder given by
>>gdal-config –datadir
In my case:
>>export GDAL_DATA=/appl/earth/gdal/2.2.1/share/gdal
Then you need to use it in external terminal.
>>xterm &

Then test.
```
>>./testall
```

4 Build and use OceanParcels on taito

Here (http://www.oceanparcels.org/) are very detailed instructions and they work well assuming you have installed miniconda2 correctly on Taito.
```
>>module load geo-env ffmpeg imagemagick
```
Install miniconda to be able to get the modules in your user local
Get it from here: https://conda.io/miniconda.html
Then install
```
>>bash Miniconda3-latest-Linux-x86_64.sh
```
Follow the instruction then when it asks the installing path type:
```
>>$HOME/.local/bin/miniconda2/
```
Get OceanParcels code from github.
```
>>git clone https://github.com/OceanParcels/parcels.git
```
Then follow the instructions from Step 2 - 3. Except that you try the example like this:
```
>>python example_peninsula.py -f 100 100
```