Using ECCO v4

Gaël Forget
Department of Earth, Atmospheric and Planetary Sciences
Massachusetts Institute of Technology

September 25, 2016

abstract

These notes pertain to the ECCO v4 state estimate, model setup, and associated codes (Forget et al., 2015). Section 1 summarizes download procedures and links to additional documentation[1]. Section 2 explains how users can proceed to re-run ECCO v4 solutions.

Contents

References

Forget, G., J.-M. Campin, P. Heimbach, C. N. Hill, R. M. Ponte, and C. Wunsch, 2015: ECCO version 4: an integrated framework for non-linear inverse modeling and global ocean state estimation. *Geoscientific Model Development*, **8 (10)**, 3071–3104, doi:10.5194/gmd-8-3071-2015, URL http://www.geosci-model-dev.net/8/3071/2015/.

Forget, G., J.-M. Campin, P. Heimbach, C. N. Hill, R. M. Ponte, and C. Wunsch, 2016: ECCO version 4: Second release. URL http://hdl.handle.net/1721.1/102062.

[1]Throughout this document links are indicated by blue colored font.

1 Downloading ECCO V4

This section first provides direction to download the ECCO v4 state estimate output (section 1.1), Matlab analysis tools (section 1.2), and MITgcm model setup (section 1.3).

1.1 ECCO V4 Output

The 'ECCO v4-release 2' state estimate for 1992-2011 (documented in Forget et al., 2016) is a minor update of 'ECCO v4-release 1' (Forget et al., 2015) that is easiest for outside users to re-run and futher benefits from a few additional corrections (listed in Forget et al., 2016). The model output for the ECCO v4–release 2 state estimate (Forget et al., 2016) is currently available via this ftp server and this opendap server from ecco-group.org. In Linux or macOS for example, a common download method is to use 'wget' at the command line by typing

```
wget --recursive ftp://mit.ecco-group.org/ecco_for_las/version_4/release2/nctiles_grid
wget --recursive ftp://mit.ecco-group.org/ecco_for_las/version_4/release2/nctiles_monthly
```

and similarly for the other directories. The 'nctiles_' directory prefix indicates that contents are provided on the native grid in 'nctiles' format, which can be read using the 'gcmfaces' Matlab toolbox (see section 1.2; Forget et al. 2015). Interpolated fields can instead be downloaded per

```
wget --recursive ftp://mit.ecco-group.org/ecco_for_las/version_4/release2/interp_monthly
```

and read directly using any 'netcdf' enabled software (e.g., Panoply in MS-Windows, Linux, or macOS). The profiles directory contains the 'MITprof' collections of collocated in situ and state estimate profiles in 'netcdf' format (Forget et al., 2015). Other directory contents are summarized in this README and specific details are provided in each the directories' README. The associated permanent archive in the harvard dataverse provides citable identifiers for the various ECCO v4–release 2 datasets (i.e., directories) that are listed in this README.pdf.

1.2 Diagnostic Tools

To analyze model output from section 1.1 or section 2.1, various tools are readily available:

- The 'gcmfaces' Matlab toolbox (Forget et al., 2015) gets installed as explained in the gcmfaces.pdf documentation. It can be used, for example, to re-generate the ECCO v4-release 2 'standard analysis' (i.e., the plots included in Forget et al. (2016)) from the section 1.1 post-processed model output or from the section 2.1 plain model output.

- The MITgcm/utils/ directory that can be downloaded via this cvs server provides Matlab and python alternatives (although for basic functionalities and plain model output only).

- Any 'netcdf' enabled software (e.g., Panoply in MS-Windows, Linux, or macOS) should also be able to read the interpolated output from section 1.1.

1.3 ECCO V4 Model

First, install the MITgcm using the MITgcm cvs server as explained in this webpage. Second, create a subdirectory called 'MITgcm/ecco_v4_r2/' and install the ECCO v4 model setup (also using the MITgcm cvs server) as follows:

```
cd MITgcm/ecco_v4_r2
cvs co -P -d global_oce_llc90 MITgcm_contrib/gael/verification/global_oce_llc90
cd global_oce_llc90/input_fields/
./gunzip_files
```

Alternatively, users can download the latest frozen versions from this webpage (MITgcm_c65y.tar.gz at this time) and this ftp server (c65z_eccov4r2.tar at this time). Re-running and verifying the ECCO v4–release 2 solution (see section 2.1) additionally requires downloading the three-hourly forcing fields (96G) and observational data (25G) inputs as follows:

```
cd MITgcm/ecco_v4_r2
wget --recursive ftp://mit.ecco-group.org/ecco_for_las/version_4/release2/input_forcing/
wget --recursive ftp://mit.ecco-group.org/ecco_for_las/version_4/release2/input_ecco/
mv mit.ecco-group.org/ecco_for_las/version_4/release2/input_forcing forcing_baseline2
mv mit.ecco-group.org/ecco_for_las/version_4/release2/input_ecco inputs_baseline2
```

2 Running ECCO V4

This section explains how the ECCO v4 setup is used to re-run the 20-year state estimate (section 2.1), other solutions (section 2.2), short regression tests (section 2.3), and an optimization example (section 2.4). As a pre-requisite, users must have downloaded the MITgcm as well as the ECCO v4 model setup and inputs (section 1.3). Based upon the section 1.3 directions, the various downloaded directories should be organized as illustrated in Fig.1 within the 'MITgcm/' directory. Running the model also requires the following software: gcc, gfortran (or alternatives), mpi (for parallel computation) and netcdf (for 'pkg/profiles'). Additional information can be found in the MITgcm howto and in the MITgcm manual.

2.1 ECCO V4–R2 Solution

Users can re-run 'ECCO v4-release 2' by following the directions in Fig. 2. The 20-year model run typically takes between 6 to 12 hours on 96 cores (depending on the computing environment). To verify the re-run results one proceeds according to Fig. 3. The expected level of accuracy for 20-year re-runs (with an up to date MITgcm; on any given computer) is reached when the displayed values are all ≤ -3 (see Forget et al., 2015, for details). The number of cores (96 by default and in Fig. 2) can be reduced to 24 by copying 'global_oce_llc90/code/SIZE.h_24cores' over 'global_oce_llc90/code/SIZE.h' before compiling the model and then running it with 'mpiexec -np 24 ./mitgcmuv'. Different compiler options (alternatives to 'linux_amd64_gfortran' in Fig. 2) are available in 'MITgcm/tools/build_options'.

```
MITgcm/
 |__model/ (core of MITgcm)
 |__pkg/ (MITgcm modules)
 |__tools/
 |   |__genmake2 (shell script)
 |   |__build_options (wrt compilers)
 |   |__...
 |__ecco_v4_r2/
 |   |__global_oce_llc90/
 |   |__forcing_baseline2/
 |   |__inputs_baseline2/
 |__...
```

Figure 1: MITgcm directory structure including the ECCO v4 directories (indicated with ”+”) downloaded according to the section 1.3 directions.

```
#1) compile the model
cd MITgcm/ecco_v4_r2/global_oce_llc90/build
../../../tools/genmake2 -mods=../code -optfile \
     ../../../tools/build_options/linux_amd64_gfortran -mpi
make depend
make -j 4

#2) link files into run directory
cd ../run
ln -s ../build/mitgcmuv .
ln -s ../input.ecco_v4/* .
ln -s ../input_fields/* .
ln -s ../../inputs_baseline2/input*/* .
ln -s ../../forcing_baseline2 .

#3) run model
mpiexec -np 96 ./mitgcmuv
```

Figure 2: Procedure to re-run the ECCO v4-release 2 solution (Forget et al., 2016)). Pre-requisites: (1) an installation of gcc, gfortran (or alternatives), and mpi; (2) an installation of the MITgcm and ECCO v4 setup (see section 1.3). The contents of ‘input.ecco_v4’ (short text files) and ‘input_fields’ (grid and other binary input) should match those found in this cvs directory. The contents of ‘forcing_baseline2’ directory should match this ftp server. The contents of ‘inputs_baseline2’ should match this ftp server. These files can readily be downloaded as explained in section 1.3.

Figure 3: Top: instructions to verify (using 'testreport_ecco.m' within Matlab) that a re-run of the ECCO v4–r2 state estimate is acceptably close to the reference result ('baseline2'). Bottom: example output from testreport_ecco.m where the re-run agrees up to 6 digits with the reference result. To activate additional tests (of meridional transports) one needs to have installed gcmfaces (see section 1.2) and uncommentrd the 'addpath' and 'gcmfaces_global' commands below (where '∼/Documents/MATLAB/gcmfaces' is meant to represent the locations where the gcmfaces toolbox were placed by the user).

```
cd MITgcm/ecco_v4_r2/global_oce_llc90
matlab -nodesktop -nodisplay

%addpath ~/Documents/MATLAB/gcmfaces;
%gcmfaces_global;

addpath results_itXX;%necessary .m and .mat files
mytest=testreport_ecco('run/');%compute the tests and display result
```

```
----------------------------------------------------------------
       &   jT &   jS &      ... &  (reference is)
run/   & (-6) & (-6) &      ...  &  baseline2
----------------------------------------------------------------
```

2.2 Other ECCO v4 Solutions

Users can also easily re-run ‘baseline1’ that most closely matches the ‘release1’ from section 1.1. To re-run ‘baseline1’ instead of ‘release2’ a few modifications to the setup are needed:

(a) get the corresponding forcing fields per

```
wget --recursive ftp://mit.ecco-group.org/ecco_for_las/version_4/release1/forcing_baseline1/
```

(b) before compiling the model: define ‘ALLOW_KAPGM_CONTROL_OLD’ and ‘ALLOW_KAPREDI_CONTROL_OLD’ in ‘global_oce_llc90/code/GMREDI_OPTIONS.h’; define ‘ALLOW_AUTODIFF_INIT_OLD’ in ‘global_oce_llc90/code/AUTODIFF_OPTIONS.h’;
(c) before running the model: copy ‘global_oce_llc90/input_itXX/data’ and ‘data.exf’ over ‘global_oce_llc90/input.ecco_v4/data’ and ‘data.exf’.

Users who may want to reproduce ‘release1’ even more precisely than ‘baseline1’ does should contact ecco-support@mit.edu to obtain additional model inputs. Users holding a TAF license can also: (a) compile the adjoint by replacing ‘make -j 4’ with ‘make adall -j 4’ in Fig. 2; (b) activate the adjoint by setting ‘useAUTODIFF=.TRUE.,’ in data.pkg; (c) run the adjoint by replacing ‘mitgcmuv’ with ‘mitgcmuv_ad’ in Fig. 2.

2.3 Short Regression Tests

To ensure continued compatibility with the up to date MITgcm, the ECCO v4 model setup is also tested on a daily basis using the ‘testreport’ command line utility (indicated in Fig.1) that compares re-runs with reference results over a few time steps (see below for guidance and the MITgcm howto for additional details). There are two versions of the setup: global_oce_cs32/ (614M) is the smaller setup that can be used for testing on a laptop, whereas global_oce_llc90/ (595M) is the larger setup that requires 12 or more processors (96 for the adjoint tests).

Instructions for their installation are provided in this README. Once installed accordingly, the smaller setup can be executed by typing:

```
cd MITgcm/verification/
./testreport -t global_oce_cs32
```

If everything proceeds as expected then the results are reported to screen as shown in Fig. 4. The daily results of the regression tests (ran on the ‘glacier’ cluster) are reported on this site. On other machines the degree of agreement (16 digits in Fig. 4) may vary and ‘testreport’ may indicate ’FAIL’. Despite the seemingly dramatic character of such message, it may not prevent reproducing 20-year solutions (see section 2.1) with acceptable accuracy. The global_oce_llc90/ experiments are too big to run using the above command (or on a laptop). With 24 processors and gfortran (these settings may differ on another machine) the adequate command instead is:

```
cd MITgcm/verification/
./testreport -of ../tools/build_options/linux_amd64_gfortran \
-j 4 -MPI 24 -command 'mpiexec -np TR_NPROC ./mitgcmuv' \
-t global_oce_llc90
```

```
default 10  ----T-----  ----S-----
G D M    c        m  s        m  s
e p a R  g  m  m  e  .  m  m  e  .
n n k u  2  i  a  a  d  i  a  a  d
2 d e n  d  n  x  n  .  n  x  n  .

Y Y Y Y>14<16 16 16 16 16 16 16 16  pass  global_oce_cs32
```

Figure 4: Abbreviated output of testreport to screen.

2.4 Adjoint And Optimization Tests

Running the adjoint tests associated with the section 2.3 requires: (1) a TAF license; (2) to soft link 'code' to 'code_ad' in global_oce_cs32/ and global_oce_llc90/. Users that hold a TAF license can further proceed with the iterative optimization test case in global_oce_cs32/input_OI/. Here the ocean model is replaced with a simple diffusion equation.

The pre-requisites are:

1. run the adjoint benchmark in global_oce_cs32/ via testreport (see section 2.3).
2. Go to MITgcm/lsopt and compile (see section 3.18 of manual).
3. Go to MITgcm/optim, replace 'natl_box_adjoint' with 'global_oce_cs32' in this Makefile, and compile as explained in section 3.18 of manual. An executable named 'optim.x' should get created in MITgcm/optim. If otherwise, please contact ecco-support@mit.edu
4. go to MITgcm/verification/global_oce_cs32/input_OI and type 'source ./prepare_run'

To match the reference results reported in this file, the user should proceed as follows

1. ./mitgcmuv_ad > output.txt
2. ./optim.x > op.txt
3. increment optimcycle by 1 in data.optim
4. go back to step #1, to run the next iteration
5. type 'grep fc costfunction000*' to display results