Downloading / Analyzing / Running ECCO Version 4 Release 2

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

November 22, 2016

abstract

The ECCO v4 r2 (Estimating the Circulation and Climate of the Ocean, version 4, release 2) state estimate covers the period from 1992 to 2011 (Forget et al., 2016). It is a minor update of the original ECCO v4 solution (Forget et al., 2015) that benefits from a few additional corrections listed in Forget et al. (2016) and is easier to analyze and re-run. Section 1 summarizes download procedures and provides links to resources that are all freely available online[1]. Section 2 provides simple directions that allow users to re-run ECCO v4 r2 in order to generate additional model output or setup their own sensitivity experiments.

Contents

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

1 Downloading ECCO Version 4

This section provides directions to download the ECCO v4 r2 output (sec. 1.1), Matlab tools to analyze the ECCO v4 r2 output (sec. 1.2), the underlying model setup (sec. 1.3) to generate more ECCO v4 r2 output (sec. 2.1), and a list of additional resources (sec. 1.4).

1.1 The Release 2 Solution

The ECCO v4 r2 state estimate output is permanently archived within the Harvard Dataverse that provides citable identifiers for the various datasets as reported in this README.pdf. For download purposes, the ECCO v4 r2 output is also made available via this ftp server by the ECCO Consortium. The various directory contents are summarized in this README and specific details are provided in each subdirectory's README. Under Linux or macOS for instance, a simple download method consists in using '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_climatology
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 LLC90 grid in the nctiles format (Forget et al., 2015) which can be read in Matlab using the gcmfaces toolbox (see section 1.2). Alternatively users can download interpolated fields, on a $1/2 \times 1/2°$ grid in the netcdf format, from the 'interp_*' directories. The 'input_*' directories contain binary and netcdf input files that can be read by MITgcm (sections 1.3 and 2.1). The profiles directory contains the MITprof collections of collocated in situ and state estimate profiles in 'netcdf' format (Forget et al., 2015).

1.2 Matlab Tools

Matlab tools are provided to analyze model output from section 1.1 or section 2.1 include:

- 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 'standard analysis' for ECCO v4 r2 (i.e., the plots included in Forget et al. (2016)) from the released model output (sec. 1.1) or from the plain, binary, model output (sec. 2.1).
- The stand-alone eccov4_lonlat.m Matlab script can be used to extract the lat-lon sector (i.e., array) of the gridded output that spans the 69°S to 56°N latitude range.

1.3 Model Setup

First, open a Linux or macOS terminal window and install the MITgcm using the MITgcm cvs server as explained in this webpage. Second, create a subdirectory called 'MITgcm/mysetups/' and install the ECCO v4 model setup in that directory using the MITgcm cvs server by typing:

```
mkdir MITgcm/mysetups
cd MITgcm/mysetups
cvs co -P -d ECCO_v4_r2 MITgcm_contrib/gael/verification/ECCO_v4_r2
```

```
cd ECCO_v4_r2/input_fields/
./gunzip_files
```

Alternatively, download the latest frozen versions from this webpage (MITgcm_c66a.tar.gz at this time) and that webpage (c66a_eccov4r2.tar at this time). Re-running and verifying the ECCO v4 r2 solution (see section 2.1) additionally requires downloading the three-hourly forcing fields (96G) and observational data (25G) inputs either from its permanent archive within the Harvard Dataverse or from the ECCO ftp server as follows:

```
cd MITgcm/mysetups/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
```

Fig. 1 provides a graphical depiction of the downloaded directories organized as is expected at the onset of section 2.1. Experienced users should feel free to re-organize directories assuming that they are comfortable with modifying the section 2.1 and Fig. 2 directions accordingly.

1.4 Other Resources

- Any netcdf enabled software (e.g., Panoply in MS-Windows, Linux, or macOS) should be able to read the interpolated output for the monthly climatology or the monthly time series.
- The state estimate fields can also be downloaded and analyzed via the NASA Sea Level Change Portal tools (https://sealevel.nasa.gov) and the Harvard Dataverse APIs (https://dataverse.harvard.edu).
- gcmfaces is also made available via github (https://github.com/gaelforget/gcmfaces)
- xmitgcm provides a python alternative (https://github.com/xgcm/xmitgcm)
- The MITgcm/utils/ directory which can be downloaded via the MITgcm cvs server and provides basic Matlab and python functionalities.
- A series of three presentations offered in May 2016 during the ECCO meeting at MIT provide an overview of the ECCO v4 r2 data sets and applications are available via researchgate.net (doi.org/10.13140/RG.2.2.33361.12647; doi.org/10.13140/RG.2.2.26650.24001; doi.org/10.13140/RG.2.2.36716.56967).

2 Running ECCO Version 4

This section explains how the ECCO version 4 setup is used to re-run the release 2 state estimate over 1992–2011 (section 2.1), other solutions (section 2.2), short regression tests (section 2.3), or an optimization test (section 2.4). As a pre-requisite, users must have downloaded MITgcm as explained in section 1.3. Running MITgcm 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.

```
MITgcm/
 |__model/ (core of MITgcm)
 |__pkg/ (MITgcm modules)
 |__tools/
 |   |__genmake2 (shell script)
 |   |__build_options (wrt compilers)
 |   |__...
 |__mysetups/ (user created)
 |   |__ECCO_v4_r2/
 |   |   |__CVS/
 |   |   |__build/
 |   |   |__code/
 |   |   |__forcing_baseline2/
 |   |   |__input/
 |   |   |__input_fields/
 |   |   |__input_itXX/
 |   |   |__inputs_baseline2/
 |   |   |__results_itXX/
 |   |   |__run/
 |   |__...
 |__...
```

Figure 1: Directory structure that includes the MITgcm as well as the ECCO v4 model setup and inputs, once they have been downloaded in 'MITgcm/mysetups' according to the section 1.3 directions, so that they can be used according to the section 2.1 and Fig. 2 directions.

2.1 The Release 2 Solution

It is here assumed that MITgcm and ECCO v4 directories have been downloaded and organized as shown in Fig. 1. Users can then re-run the ECCO version 4 release 2 solution by following the directions in Fig. 2. Afterwards they are strongly encouraged to verify their results by using the included testreport_ecco.m Matlab script as depicted in Fig. 3. The expected level of accuracy for 20-year re-runs, based upon an up-to-date MITgcm code and a standard computing environment, is reached when the displayed values are all ≤ -3. Interpretation of the testreport_ecco.m output is explained in detail in Forget et al. (2015).

The 20-year model run typically takes between 6 to 12 hours of wall-clock time on 96 cores using a modern computing environment. The number of cores is 96 by default as reflected by Fig. 2 but can be reduced to 24 simply by copying 'ECCO_v4_r2/code/SIZE.h_24cores' over 'ECCO_v4_r2/code/SIZE.h' before compiling the model and then running it with '-np 24' rather than '-np 96' in Fig. 2. However, it should be noted that reducing the number of cores increases wall-clock time and memory requirements.

```
#1) compile model
cd MITgcm/mysetups/ECCO_v4_r2/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/* .
ln -s ../input_fields/* .
ln -s ../inputs_baseline2/input*/* .
ln -s ../forcing_baseline2 .

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

Figure 2: Procedure to compile MITgcm and re-run the ECCO v4 r2 solution (Forget et al., 2016). Prerequisites: (1) gcc, gfortran (or alternatives), mpi (for parallel computation) and netcdf (for pkg/profiles); (2) MITgcm and ECCO v4 setup (see section 1.3); (3) input directories organized as shown in Fig. 1 (see section 1.3). Other compiler options, besides linux_amd64_gfortran, are provided by the MITgcm development team in 'MITgcm/tools/build_options/' for cases when gfortran is not available. The contents of 'input/' (text files) and 'input_fields/' (binary files) should match those found in this cvs directory. Note: the 'input_fields/' contents must be de-compressed once. The contents of 'forcing_baseline2/' directory should match this ftp server. The contents of 'inputs_baseline2/' should match this ftp server. These directories can readily be downloaded as explained in section 1.3.

Figure 3: Top: instructions to gauge the accuracy of a re-run of ECCO v4 r2 (Forget et al., 2016) using the testreport_ecco.m Matlab script (Forget et al., 2015). Bottom: sample output of testreport_ecco.m where the re-run agrees up to 6 digits with the reference result. Additional tests of meridional transports can be activated by users who have installed the gcmfaces toolbox (Forget et al., 2015) as explained in section 1.2. To this end, users would uncomment the 'addpath ∼/Documents/MATLAB/gcmfaces;' and 'gcmfaces_global;' commands below and, if needed, replace '∼/Documents/MATLAB/gcmfaces' with the location where gcmfaces has been installed.

```
cd MITgcm/mysetups/ECCO_v4_r2
matlab -nodesktop -nodisplay

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

addpath results_itXX;%add necessary .m and .mat files to path
mytest=testreport_ecco('run/');%compute tests and display results
```

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

2.2 Other Solutions

It is here assumed that MITgcm and ECCO v4 directories have been downloaded and organized as shown in Fig. 1. Users can then re-run the 'baseline 1' solution that more closely matches the original, release 1, solution of Forget et al. (2015). However, to re-run baseline 1 instead of release 2, a few modifications to the setup are needed:

(a) download the corresponding forcing fields as follows:

```
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 'ECCO_v4_r2/code/GMREDI_OPTIONS.h'; define 'ALLOW_AUTODIFF_INIT_OLD' in 'ECCO_v4_r2/code/AUTODIFF_OPTIONS.h';
(c) before running the model: copy 'ECCO_v4_r2/input_itXX/data' and 'data.exf' over 'ECCO_v4_r2/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 Forward 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 MITgcm's 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). These tests use dedicated versions of the ECCO v4 model setup which are located within MITgcm_contrib/verification_other/.

global_oce_llc90/ (595M) uses the same LLC90 grid as the production ECCO v4 setup does (section 2.1). Users are advised against running forward tests using fewer than 12 cores (96 for adjoint tests) to avoid potential memory overloads. global_oce_cs32/ (614M) uses the much coarser resolution CS32 grid and can thus be used on any modern laptop. Instructions for their installation are provided in this README and that README, respectively. Once installed, the smaller setup for instance can be executed on one core 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'. Note: despite the seemingly dramatic character of this message, users may still be able to reproduce 20-year solutions with acceptable accuracy (section 2.1). To test global_oce_llc90/ using 24 processors and gfortran the corresponding command typically 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 example of testreport output to screen.

2.4 Other Short Tests

Running the adjoint tests associated with section 2.3 requires: (1) a TAF license; (2) to soft link ‘code’ as ‘code_ad’ in global_oce_cs32/ and global_oce_llc90/. Users that hold a TAF license can then 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 mitgcm-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, users 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

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.