

# Using ECCO v4

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

March 2, 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<sup>1</sup>. Section 2 explains how ECCO v4 solutions, or corresponding short regression tests, can be re-run.

## Contents

|          |                                            |          |
|----------|--------------------------------------------|----------|
| <b>1</b> | <b>Downloading ECCO v4</b>                 | <b>2</b> |
| 1.1      | Released ECCO v4 Solution . . . . .        | 2        |
| 1.2      | Diagnostic Tools . . . . .                 | 2        |
| 1.3      | ECCO v4 setup . . . . .                    | 2        |
| <b>2</b> | <b>Running ECCO v4</b>                     | <b>3</b> |
| 2.1      | Baseline ECCO v4 solution . . . . .        | 3        |
| 2.2      | Other ECCO v4 Solutions . . . . .          | 6        |
| 2.3      | Short Regression Tests . . . . .           | 6        |
| 2.4      | Iterative Optimization Test Case . . . . . | 7        |

## 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/>.

---

<sup>1</sup>Throughout this document links are indicated by blue colored font.

# 1 Downloading ECCO v4

This section first provides direction to download the ECCO v4-release 1 state estimate output (section 1.1) and associated matlab analysis tools (section 1.2). It then explains download procedures for the ECCO v4 model setup and MITgcm (section 1.3). Sections 1.2 and 1.3 rely on [the MITgcm cvs server](#) (see section 1.3). As an alternative, frozen versions are available at [this site](#).

## 1.1 Released ECCO v4 Solution

The model output for the ECCO v4-release 1 state estimate is available via [this opendap server](#) and [this ftp server](#) from [ecco-group.org](#). The servers provide the [grid files](#) and [monthly output fields](#) in ‘nctiles’ format, as well as [collocated in situ and state estimate profiles](#) in ‘MITprof’ format. The ‘nctiles’ and ‘MITprof’ format are described in Forget et al. (2015). The files can be downloaded at the command line, e.g. within a linux environment, by typing

```
wget --recursive ftp://mit.ecco-group.org/ecco_for_las/version_4/release1/nctiles_grid
wget --recursive ftp://mit.ecco-group.org/ecco_for_las/version_4/release1/nctiles
wget --recursive ftp://mit.ecco-group.org/ecco_for_las/version_4/release1/MITprof_release1
```

## 1.2 Diagnostic Tools

To analyze model output from section 1.1 or section 2.1, two sets of Matlab tools are available:

- The [gcmfaces+MITprof](#) framework (see Forget et al., 2015) normally gets installed using [this shell script](#) (via [cvs](#)). Alternatively, frozen versions are available at [this site](#).
- Basic MITgcm [tools](#) can also be downloaded via [cvs](#).

For example, one may readily regenerate [this set of diagnostics](#) for the ECCO v4-release 1 state estimate using ‘diags\_driver.m’ as explained in [the gcmfaces.pdf documentation](#).

## 1.3 ECCO v4 setup

First, install the MITgcm using cvs as explained at [this site](#). Second, install the ECCO v4 model setup on the LLC90 and CS32 grids (see Forget et al., 2015) also via [cvs](#) according to:

```
cd MITgcm/verification
cvs co -P -d global_oce_llc90 MITgcm_contrib/gael/verification/global_oce_llc90
cvs co -P -d global_oce_cs32 MITgcm_contrib/gael/verification/global_oce_cs32
cd global_oce_llc90/input_fields/
./gunzip_files
cd ../../
```

Alternatively, frozen versions are available at [this site](#). [global\\_oce\\_cs32/](#) (614M) is a small setup used only for testing, whereas [global\\_oce\\_llc90/](#) (595M) is the production setup that typically runs on 96 processors. It is advised to always download or update the two setups together since they are tied to each other for the purpose of section 2.3 even though only [global\\_oce\\_llc90/](#)

is needed for the purpose of section 2.1. Running and verifying the section 2.1 solutions furthermore requires downloading the three-hourly forcing fields (96G) and in-situ data inputs (7G) from:

```
cd MITgcm/verification
wget --recursive ftp://mit.ecco-group.org/ecco_for_las/version_4/forcing_baseline2/
wget --recursive ftp://mit.ecco-group.org/ecco_for_las/version_4/inputs_baseline2/
mv mit.ecco-group.org/ecco_for_las/version_4/forcing_baseline2 .
mv mit.ecco-group.org/ecco_for_las/version_4/inputs_baseline2 .
```

Running the section 2.3 regression tests instead requires ‘global\_oce\_input\_fields/’ (1.6G):

```
cd MITgcm/verification
wget http://mitgcm.org/~gforget/global_oce_input_fields.tar.gz
gunzip global_oce_input_fields.tar.gz
tar xf global_oce_input_fields.tar
rm -f global_oce_input_fields.tar
```

## 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) or short regression tests (section 2.3). As a pre-requisite, one 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 MITgcm furthermore requires the following software: gcc, gfortran (or alternatives), mpi (only for parallel runs) and netcdf (only if ‘pkg/profiles’ is used). Additional information can be found in [the MITgcm howto](#) and in [the MITgcm manual](#).

### 2.1 Baseline ECCO v4 solution

The ‘baseline2’ state estimate for 1992-2011 is a minor update of ‘ECCO v4-release1’ (Forget et al., 2015) that is easiest for outside users to re-run. It further benefits from a few additional corrections: 1) inclusion of geothermal heating at the sea floor; 2) re-inclusion of targeted bottom viscosity; 3) re-inclusion of estimated wind stress forcing adjustments over 2000-2011; 4) re-adjustment of global mean precipitation to match the aviso global mean sea level time series. Discussion of these specific aspects of the solution can be found in Forget et al. (2015). The standard analysis document for ‘baseline2’ is available [here](#)<sup>2</sup>.

To re-run ‘baseline2’ one proceeds according to Fig. 2. A 20-year ECCO v4 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  $\leq -3$  (see Forget et al., 2015, for details).

The number of cores (96 by default and in Fig. 2) can be reduced to, e.g., 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’.

---

<sup>2</sup>coming soon...

```

MITgcm/
├── model/ (core of MITgcm)
├── pkg/ (MITgcm modules)
├── verification/
│   ├── testreport (shell script)
│   ├── aim.5l_cs (mitgcm regression test)
│   ├── + global_oce_cs32/ (for laptops)
│   ├── + global_oce_llc90/ (for computers)
│   ├── + global_oce_input_fields/ (inputs)
│   ├── + forcing_baseline2/ (inputs)
│   ├── + inputs_baseline2/ (inputs)
│   ├── hs94.128x64x5 (mitgcm regression test)
│   └── ...
└── tools/
    ├── genmake2 (shell script)
    ├── build_options (wrt compilers)
    └── ...

```

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/verification/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 state estimate (currently the ‘baseline2’ version). Pre-requisites: (1) installation of gcc, gfortran (or alternatives), and mpi (only for parallel runs); (2) installation of the MITgcm and ECCO v4 setup installation according to section 1.3. The contents of ‘input.ecco\_v4’ (short text files) and ‘input\_fields’ (grid and other binary input) should match [this site](#). The contents of ‘forcing\_baseline2’ directory should match [this site](#). The contents of ‘inputs\_baseline2’ should match [this site](#). These files can be downloaded freely, e.g., 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 state estimate (currently the ‘baseline2’ version) is acceptably close to the reference result. 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 uncommented the ‘addpath’ and ‘gcmfaces\_global’ commands below (where ‘ /Documents/MATLAB/gcmfaces’ is a user specific path).

```
cd MITgcm/verification/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
-----
```



It should be stressed that the bigger ECCO v4 setup ([global\\_oce\\_llc90/](#)) requires at least 12 cores in forward mode (96 in adjoint mode) and therefore should not be run using the above command (or on a laptop). Instead the [global\\_oce\\_llc90/](#) regression tests use mpi:

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

with 24 processors and gfortran (these settings may differ on another machine).

To prevent users from inadvertently running the llc90 tests in serial mode (e.g. via a './testreport' call) the results were moved from their expected location to 'global\_oce\_llc90/results/hidden/'. To activate the llc90 tests, the 'output\*' files contained in this directory must therefore be soft linked to 'global\_oce\_llc90/results/'. To further activate the adjoint tests (which require a [TAF](#) license) one needs to soft link 'code' to 'code\_ad' in [global\\_oce\\_cs32/](#) and [global\\_oce\\_llc90/](#).

To slightly reduce memory and storage needs in these short regression tests, one can copy 'global\_oce\_llc90/code/tamc.h\_short' and 'global\_oce\_llc90/code/PROFILES\_SIZE.h\_short' over 'global\_oce\_llc90/code/tamc.h' and 'global\_oce\_llc90/code/PROFILES\_SIZE.h'. It should also be noted that the llc90 input files in 'global\_oce\_llc90/input\_fields/' need to be un-compressed (using 'gunzip\_files'; see section [1.3](#)) for the regression tests to proceed as expected.

## 2.4 Iterative Optimization Test Case

The [global\\_oce\\_cs32/input\\_OI](#) directory implements an iterative optimization test case. It boils down to optimal interpolation solved by a variational method using the MITgcm adjoint (the ocean model being replaced with a simple diffusion equation here). 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'

Then the iterative optimization itself proceeds 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 (Fig. [5](#)).

|                        |                    |            |
|------------------------|--------------------|------------|
| costfunction0000: fc = | 4118.1987222194211 | 0.00000000 |
| costfunction0001: fc = | 1523.9310891186672 | 0.00000000 |
| costfunction0002: fc = | 1053.3611790049420 | 0.00000000 |
| costfunction0003: fc = | 790.10479375339185 | 0.00000000 |

Figure 5: Results of iterative optimization after 3 iterations carried out as explained in section [2.4](#).