Introduction to the practical exercises

I-Han Chen

ihanchen@ucar.edu

NCAR | MESOSCALE & MICROSCALE METEOROLOGY

Instructions for MPAS-JEDI practice exercise

https://www2.mmm.ucar.edu/projects/mpas-jedi/tutorial/202310NCU/

Taiwania-3 HPC system

```
ssh -Y [account]@twnia3.nchc.org.tw
```

Check what shell you are using

- echo $SHELL

If yours are not under bash

- bash

We will submit job scripts to a batch queueing system to run on computing nodes.

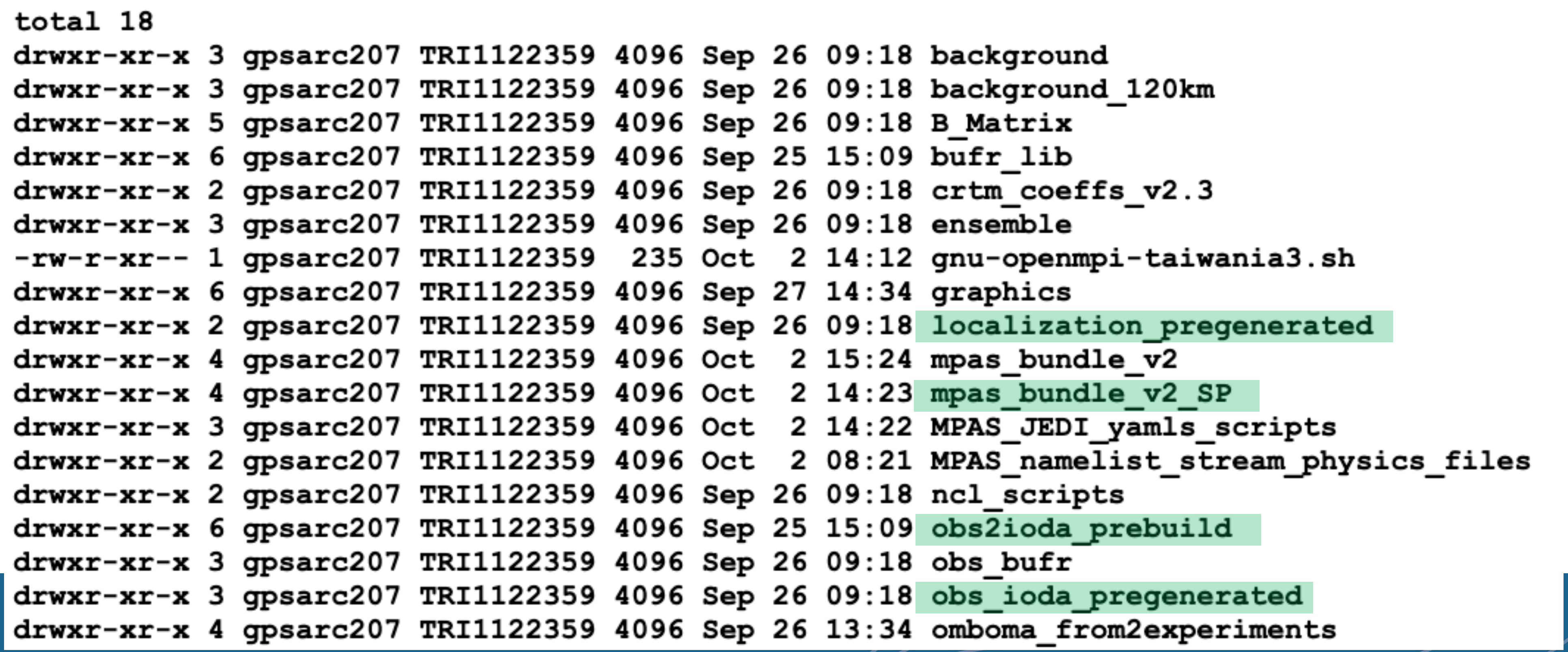
Avoid running any compute-intensive on the* *“LOGIN nodes”***

sbatch	Submit a job script
squeue -u $USER	Check the status of your pending and running jobs
scancel	Delete a queued or running job

Obtain the mpas-jedi-tutorial folder

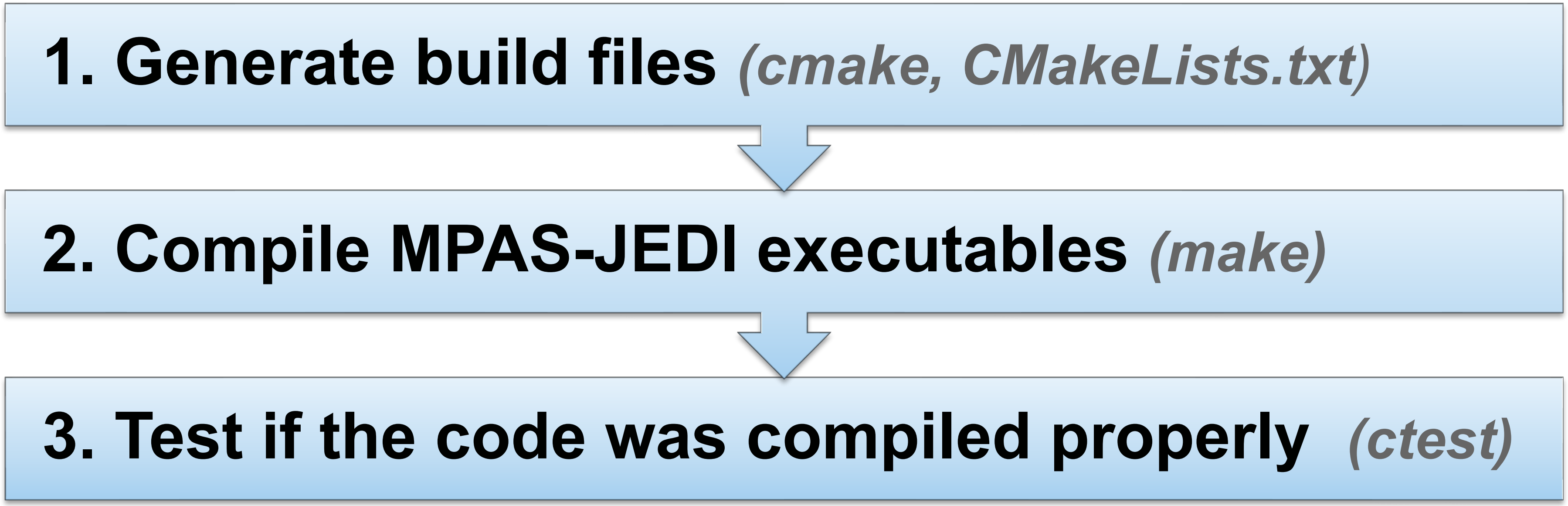
- **cd /work/$USER**
- **cp -r /work/gpsarc207/mpas_jedi_tutorial .**

  (this will create your own working directory that contains prebuild codes-> /work/$USER/mpas_jedi_tutorial)
- **ls -l mpas_jedi_tutorial**, *you will see:*



```
total 18
drwxr-xr-x 3 gpsarc207 TRI1122359 4096 Sep 26 09:18 background
drwxr-xr-x 3 gpsarc207 TRI1122359 4096 Sep 26 09:18 background_120km
drwxr-xr-x 5 gpsarc207 TRI1122359 4096 Sep 26 09:18 B_Matrix
drwxr-xr-x 6 gpsarc207 TRI1122359 4096 Sep 25 15:09 bufr_lib
drwxr-xr-x 2 gpsarc207 TRI1122359 4096 Sep 26 09:18 crtm_coeffs_v2.3
drwxr-xr-x 3 gpsarc207 TRI1122359 4096 Sep 26 09:18 ensemble
-rw-r-xr-- 1 gpsarc207 TRI1122359  235 Oct  2 14:12 gnu-openmpi-taiwania3.sh
drwxr-xr-x 6 gpsarc207 TRI1122359 4096 Sep 27 14:34 graphics
drwxr-xr-x 2 gpsarc207 TRI1122359 4096 Sep 26 09:18 localization_pregenerated
drwxr-xr-x 4 gpsarc207 TRI1122359 4096 Oct  2 15:24 mpas_bundle_v2
drwxr-xr-x 4 gpsarc207 TRI1122359 4096 Oct  2 14:23 mpas_bundle_v2_SP
drwxr-xr-x 3 gpsarc207 TRI1122359 4096 Oct  2 14:22 MPAS_JEDI_yamls_scripts
drwxr-xr-x 2 gpsarc207 TRI1122359 4096 Oct  2 08:21 MPAS_namelist_stream_physics_files
drwxr-xr-x 2 gpsarc207 TRI1122359 4096 Sep 26 09:18 ncl_scripts
drwxr-xr-x 6 gpsarc207 TRI1122359 4096 Sep 25 15:09 obs2ioda_prebuild
drwxr-xr-x 3 gpsarc207 TRI1122359 4096 Sep 26 09:18 obs_bufr
drwxr-xr-x 3 gpsarc207 TRI1122359 4096 Sep 26 09:18 obs_ioda_pregenerated
drwxr-xr-x 4 gpsarc207 TRI1122359 4096 Sep 26 13:34 omboma_from2experiments
```

Build the MPAS-JEDI and its dependencies



1. Generate build files *(cmake, CMakeLists.txt)*

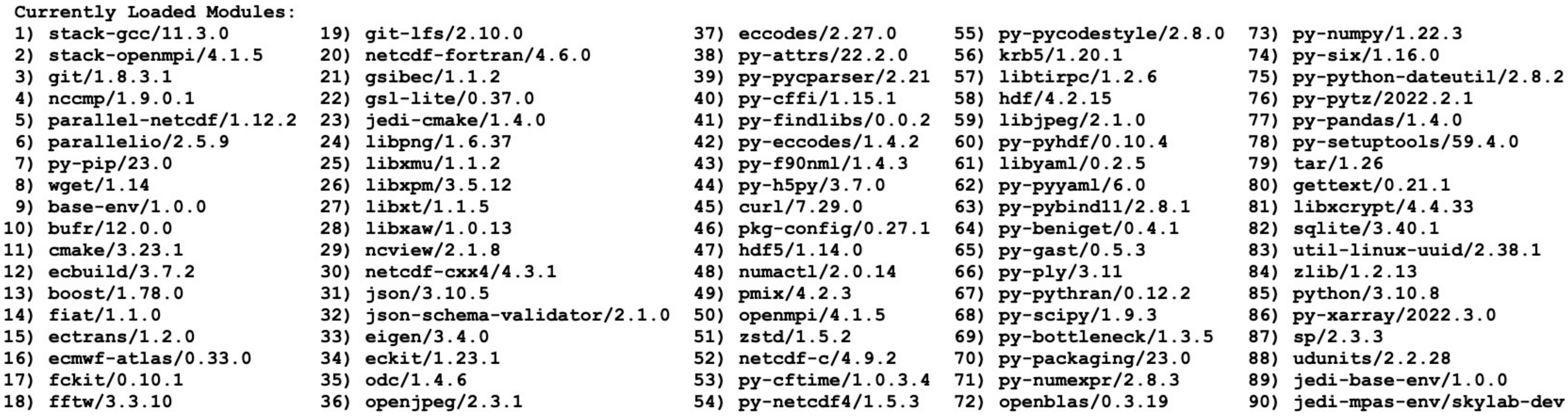
2. Compile MPAS-JEDI executables *(make)*

3. Test if the code was compiled properly *(ctest)*

Required spack-stack build environment

This tutorial does not cover the installation of spack-stack, which was pre-installed on Taiwania-3.

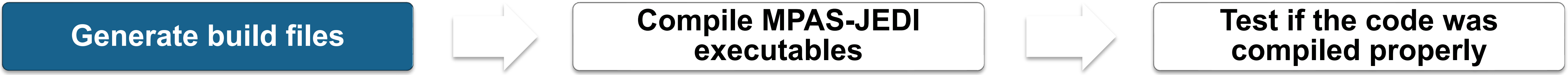
- source ../../gnu-openmpi-taiwania3.sh
- module list



```
 Currently Loaded Modules:
  1) stack-gcc/11.3.0          19) git-lfs/2.10.0                 37) eccodes/2.27.0        55) py-pycodestyle/2.8.0   73) py-numpy/1.22.3
  2) stack-openmpi/4.1.5       20) netcdf-fortran/4.6.0           38) py-attrs/22.2.0       56) krb5/1.20.1            74) py-six/1.16.0
  3) git/1.8.3.1               21) gsibec/1.1.2                   39) py-pycparser/2.21     57) libtirpc/1.2.6         75) py-python-dateutil/2.8.2
  4) nccmp/1.9.0.1             22) gsl-lite/0.37.0                40) py-cffi/1.15.1        58) hdf/4.2.15             76) py-pytz/2022.2.1
  5) parallel-netcdf/1.12.2    23) jedi-cmake/1.4.0               41) py-findlibs/0.0.2     59) libjpeg/2.1.0          77) py-pandas/1.4.0
  6) parallelio/2.5.9          24) libpng/1.6.37                  42) py-eccodes/1.4.2      60) py-pyhdf/0.10.4        78) py-setuptools/59.4.0
  7) py-pip/23.0               25) libxmu/1.1.2                   43) py-f90nml/1.4.3       61) libyaml/0.2.5          79) tar/1.26
  8) wget/1.14                 26) libxpm/3.5.12                  44) py-h5py/3.7.0         62) py-pyyaml/6.0          80) gettext/0.21.1
  9) base-env/1.0.0            27) libxt/1.1.5                    45) curl/7.29.0           63) py-pybind11/2.8.1      81) libxcrypt/4.4.33
 10) bufr/12.0.0               28) libxaw/1.0.13                  46) pkg-config/0.27.1     64) py-beniget/0.4.1       82) sqlite/3.40.1
 11) cmake/3.23.1              29) ncview/2.1.8                   47) hdf5/1.14.0           65) py-gast/0.5.3          83) util-linux-uuid/2.38.1
 12) ecbuild/3.7.2             30) netcdf-cxx4/4.3.1              48) numactl/2.0.14        66) py-ply/3.11            84) zlib/1.2.13
 13) boost/1.78.0              31) json/3.10.5                    49) pmix/4.2.3            67) py-pythran/0.12.2      85) python/3.10.8
 14) fiat/1.1.0                32) json-schema-validator/2.1.0    50) openmpi/4.1.5         68) py-scipy/1.9.3         86) py-xarray/2022.3.0
 15) ectrans/1.2.0             33) eigen/3.4.0                    51) zstd/1.5.2            69) py-bottleneck/1.3.5    87) sp/2.3.3
 16) ecmwf-atlas/0.33.0        34) eckit/1.23.1                   52) netcdf-c/4.9.2        70) py-packaging/23.0      88) udunits/2.2.28
 17) fckit/0.10.1              35) odc/1.4.6                      53) py-cftime/1.0.3.4     71) py-numexpr/2.8.3       89) jedi-base-env/1.0.0
 18) fftw/3.3.10               36) openjpeg/2.3.1                 54) py-netcdf4/1.5.3      72) openblas/0.3.19        90) jedi-mpas-env/skylab-dev
```

NCAR
UCAR

More information (https://github.com/JCSDA/spack-stack)



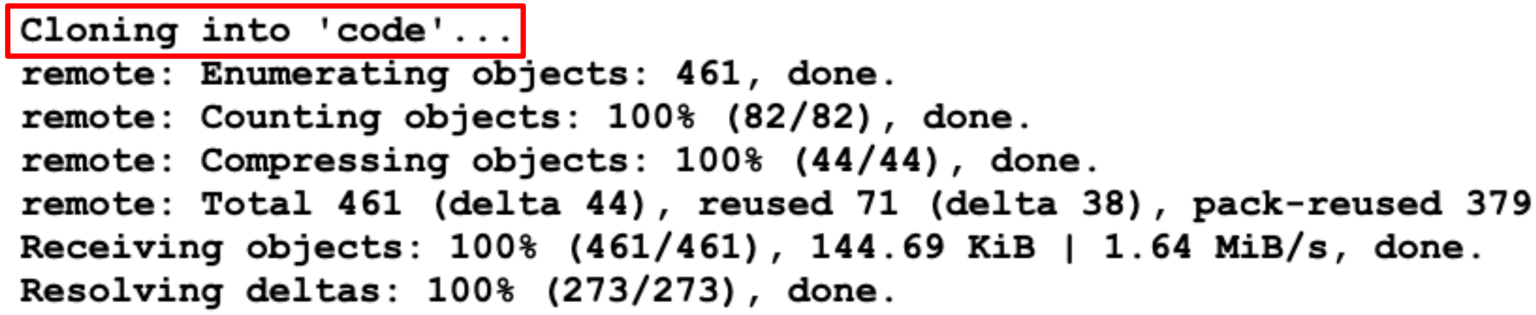
Generate build files

Compile MPAS-JEDI executables

Test if the code was compiled properly

Clone mpas-bundle repository and checkout the ‘release/2.0.0’ branch

- cd /work/$USER/mpas_jedi_tutorial
- mkdir mpas_bundle_v2 ; cd mpas_bundle_v2
- git clone -b release/2.0.0 https://github.com/JCSDA/mpas-bundle.git code



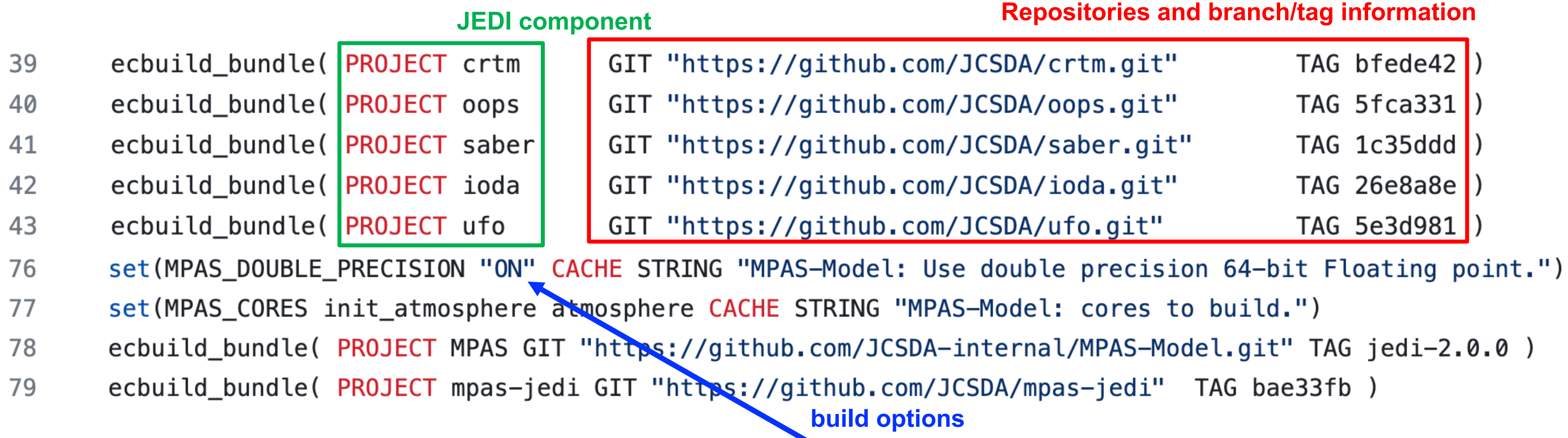
```
Cloning into 'code'...
remote: Enumerating objects: 461, done.
remote: Counting objects: 100% (82/82), done.
remote: Compressing objects: 100% (44/44), done.
remote: Total 461 (delta 44), reused 71 (delta 38), pack-reused 379
Receiving objects: 100% (461/461), 144.69 KiB | 1.64 MiB/s, done.
Resolving deltas: 100% (273/273), done.
```

The mpas-bundle repository does not contain actual source code. Instead, you will obtain the CMakeLists.txt file under code.

Generate build files → Compile MPAS-JEDI executables → Test if the code was compiled properly

CMakeLists.txt tells CMake system how to compile and build the code

- **vi code/CMakeLists.txt**



JEDI component

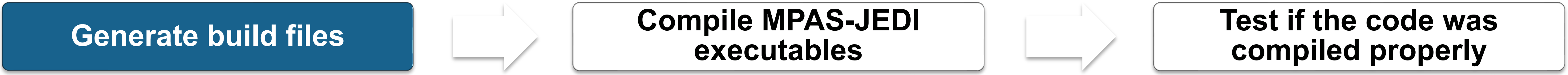
Repositories and branch/tag information

build options

```
39    ecbuild_bundle( PROJECT crtm     GIT "https://github.com/JCSDA/crtm.git"       TAG bfede42 )
40    ecbuild_bundle( PROJECT oops     GIT "https://github.com/JCSDA/oops.git"       TAG 5fca331 )
41    ecbuild_bundle( PROJECT saber    GIT "https://github.com/JCSDA/saber.git"      TAG 1c35ddd )
42    ecbuild_bundle( PROJECT ioda     GIT "https://github.com/JCSDA/ioda.git"       TAG 26e8a8e )
43    ecbuild_bundle( PROJECT ufo      GIT "https://github.com/JCSDA/ufo.git"        TAG 5e3d981 )
76    set(MPAS_DOUBLE_PRECISION "ON" CACHE STRING "MPAS-Model: Use double precision 64-bit Floating point.")
77    set(MPAS_CORES init_atmosphere atmosphere CACHE STRING "MPAS-Model: cores to build.")
78    ecbuild_bundle( PROJECT MPAS GIT "https://github.com/JCSDA-internal/MPAS-Model.git" TAG jedi-2.0.0 )
79    ecbuild_bundle( PROJECT mpas-jedi GIT "https://github.com/JCSDA/mpas-jedi"  TAG bae33fb )
```

Note1: MPAS-Model inside mpas-bundle is built in double precision by default, it is suggested to build it in single precision for production by changing Line76 from “ON” to “OFF”.

Note2: We will use the pre-build single-precision mpas-bundle executable in practical sessions



Generate build files

Compile MPAS-JEDI executables

Test if the code was compiled properly

Use cmake to generate build files

- **mkdir build ; cd build**

  (We will compile the executables under build)

- **cmake ../code**
  - *git clone repos in CMakeLists.txt into ../code*
  - *generate makefiles under build*

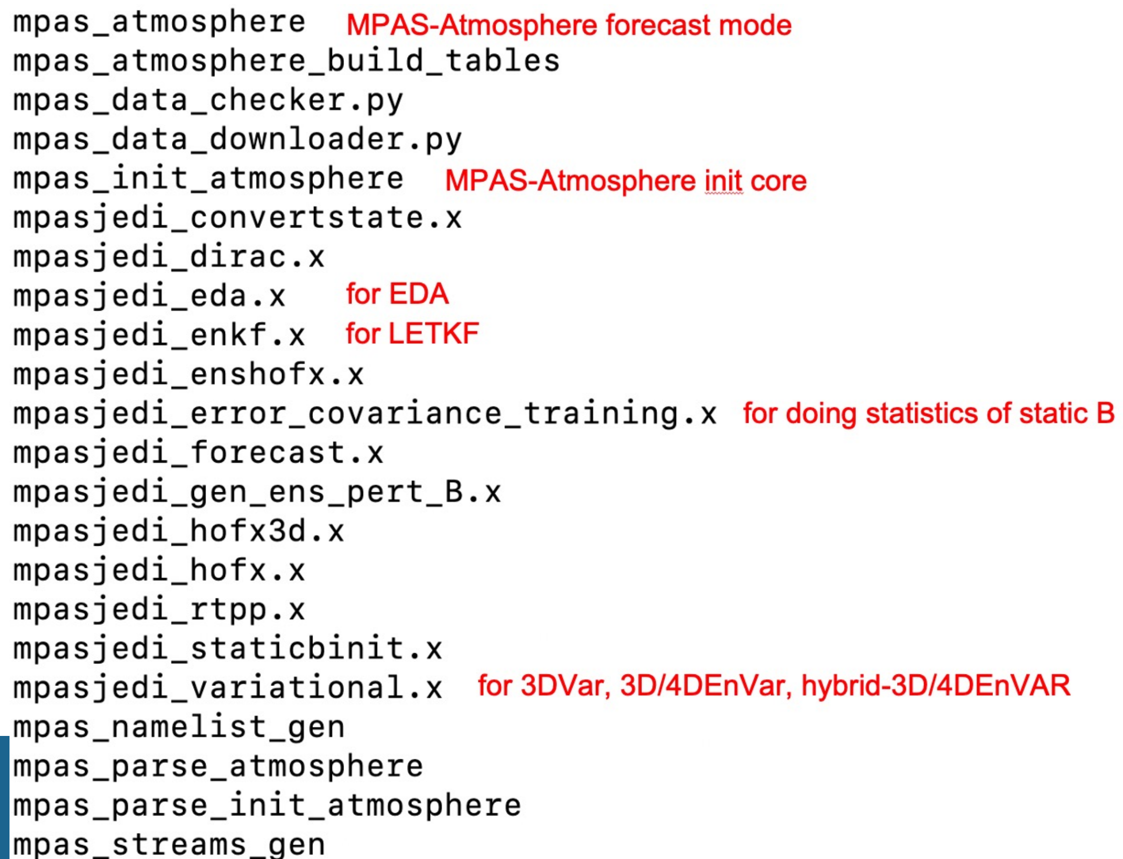
Generate build files → **Compile MPAS-JEDI executables** → Test if the code was compiled properly

Compile MPAS-JEDI executables

- **make -j14**

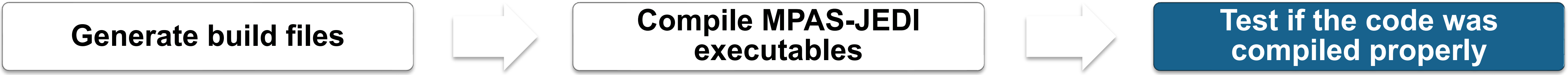
  (compile MPAS-JEDI using a login node with 14 cores)

  (The compilation will take ~14 min to complete)



```
mpas_atmosphere    MPAS-Atmosphere forecast mode
mpas_atmosphere_build_tables
mpas_data_checker.py
mpas_data_downloader.py
mpas_init_atmosphere    MPAS-Atmosphere init core
mpasjedi_convertstate.x
mpasjedi_dirac.x
mpasjedi_eda.x     for EDA
mpasjedi_enkf.x    for LETKF
mpasjedi_enshofx.x
mpasjedi_error_covariance_training.x  for doing statistics of static B
mpasjedi_forecast.x
mpasjedi_gen_ens_pert_B.x
mpasjedi_hofx3d.x
mpasjedi_hofx.x
mpasjedi_rtpp.x
mpasjedi_staticbinit.x
mpasjedi_variational.x  for 3DVar, 3D/4DEnVar, hybrid-3D/4DEnVAR
mpas_namelist_gen
mpas_parse_atmosphere
mpas_parse_init_atmosphere
mpas_streams_gen
```

MPAS-JEDI related executables under ~build/bin


Generate build files

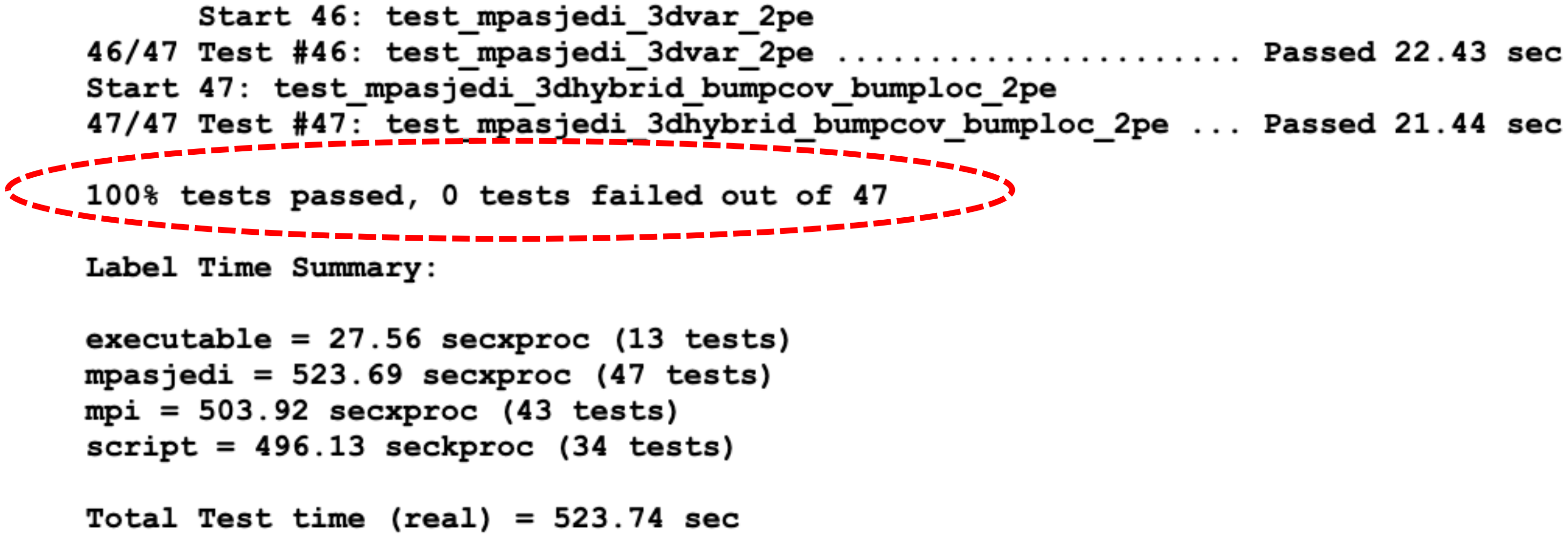
Compile MPAS-JEDI executables

Test if the code was compiled properly

Use ctest to ensure that the code was compiled properly

- cd mpas-jedi
- ctest

(take ~8 min to finish)

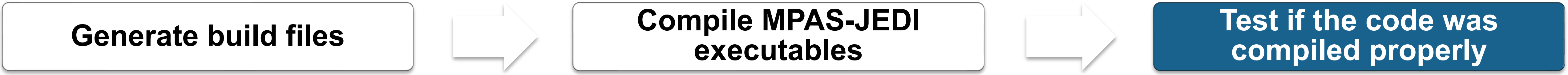

```
         Start 46: test_mpasjedi_3dvar_2pe
46/47 Test #46: test_mpasjedi_3dvar_2pe ...................... Passed 22.43 sec
Start 47: test_mpasjedi_3dhybrid_bumpcov_bumploc_2pe
47/47 Test #47: test_mpasjedi_3dhybrid_bumpcov_bumploc_2pe ... Passed 21.44 sec

100% tests passed, 0 tests failed out of 47

Label Time Summary:

executable = 27.56 secxproc (13 tests)
mpasjedi = 523.69 secxproc (47 tests)
mpi = 503.92 secxproc (43 tests)
script = 496.13 seckproc (34 tests)

Total Test time (real) = 523.74 sec
```



Generate build files

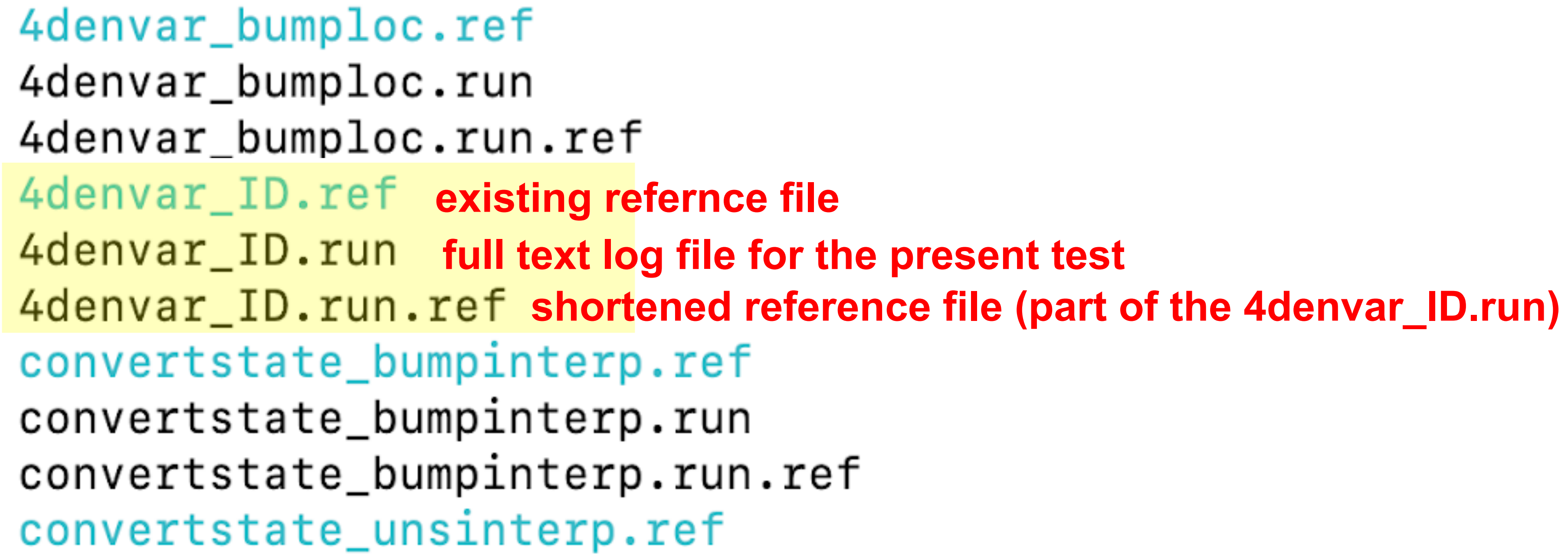
Compile MPAS-JEDI executables

Test if the code was compiled properly

What a ctest case ‘Passed’ means?

Each test run will produce text log files

(Under ~mpas_bundle/build/mpas-jedi/test/testoutput**)**



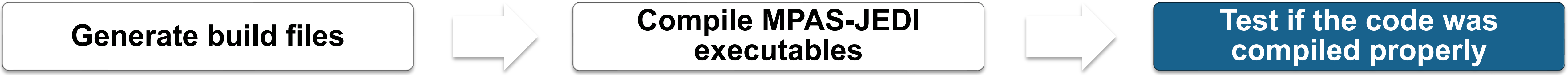
```
4denvar_bumploc.ref
4denvar_bumploc.run
4denvar_bumploc.run.ref
4denvar_ID.ref
4denvar_ID.run
4denvar_ID.run.ref
convertstate_bumpinterp.ref
convertstate_bumpinterp.run
convertstate_bumpinterp.run.ref
convertstate_unsinterp.ref
```

existing refernce file

full text log file for the present test

shortened reference file (part of the 4denvar_ID.run)

- **4dvar_ID.run.ref is compared with the existing 4dvar_ID.ref.**
- **The test is deemed as “Passed” if numerical values between the two files are identical or within a tolerance,.**

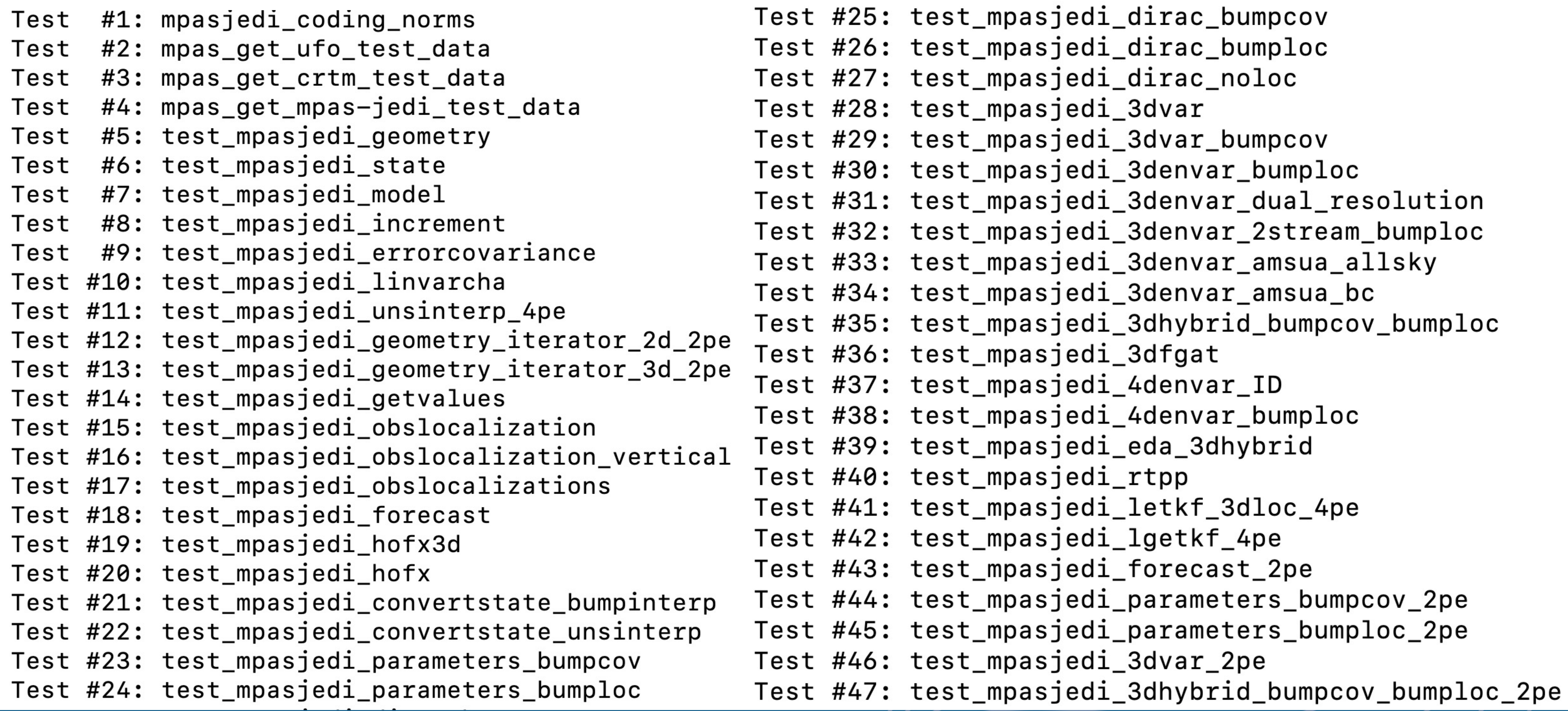


Generate build files

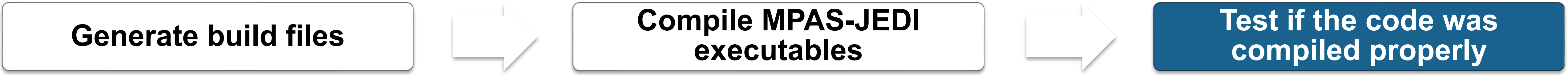
Compile MPAS-JEDI executables

Test if the code was compiled properly

‘ctest –N’ will list, but not run 47 test cases



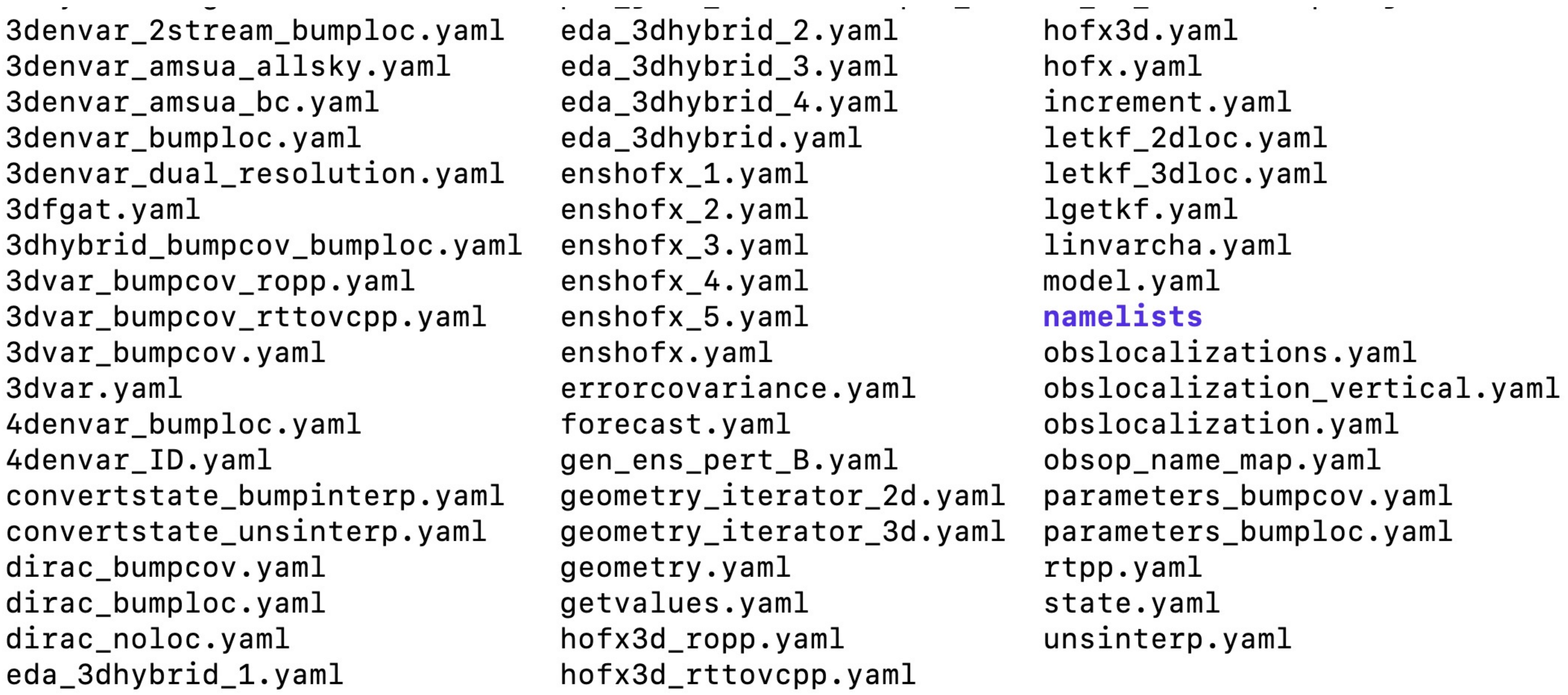
```
Test  #1: mpasjedi_coding_norms
Test  #2: mpas_get_ufo_test_data
Test  #3: mpas_get_crtm_test_data
Test  #4: mpas_get_mpas-jedi_test_data
Test  #5: test_mpasjedi_geometry
Test  #6: test_mpasjedi_state
Test  #7: test_mpasjedi_model
Test  #8: test_mpasjedi_increment
Test  #9: test_mpasjedi_errorcovariance
Test #10: test_mpasjedi_linvarcha
Test #11: test_mpasjedi_unsinterp_4pe
Test #12: test_mpasjedi_geometry_iterator_2d_2pe
Test #13: test_mpasjedi_geometry_iterator_3d_2pe
Test #14: test_mpasjedi_getvalues
Test #15: test_mpasjedi_obslocalization
Test #16: test_mpasjedi_obslocalization_vertical
Test #17: test_mpasjedi_obslocalizations
Test #18: test_mpasjedi_forecast
Test #19: test_mpasjedi_hofx3d
Test #20: test_mpasjedi_hofx
Test #21: test_mpasjedi_convertstate_bumpinterp
Test #22: test_mpasjedi_convertstate_unsinterp
Test #23: test_mpasjedi_parameters_bumpcov
Test #24: test_mpasjedi_parameters_bumploc
Test #25: test_mpasjedi_dirac_bumpcov
Test #26: test_mpasjedi_dirac_bumploc
Test #27: test_mpasjedi_dirac_noloc
Test #28: test_mpasjedi_3dvar
Test #29: test_mpasjedi_3dvar_bumpcov
Test #30: test_mpasjedi_3denvar_bumploc
Test #31: test_mpasjedi_3denvar_dual_resolution
Test #32: test_mpasjedi_3denvar_2stream_bumploc
Test #33: test_mpasjedi_3denvar_amsua_allsky
Test #34: test_mpasjedi_3denvar_amsua_bc
Test #35: test_mpasjedi_3dhybrid_bumpcov_bumploc
Test #36: test_mpasjedi_3dfgat
Test #37: test_mpasjedi_4denvar_ID
Test #38: test_mpasjedi_4denvar_bumploc
Test #39: test_mpasjedi_eda_3dhybrid
Test #40: test_mpasjedi_rtpp
Test #41: test_mpasjedi_letkf_3dloc_4pe
Test #42: test_mpasjedi_lgetkf_4pe
Test #43: test_mpasjedi_forecast_2pe
Test #44: test_mpasjedi_parameters_bumpcov_2pe
Test #45: test_mpasjedi_parameters_bumploc_2pe
Test #46: test_mpasjedi_3dvar_2pe
Test #47: test_mpasjedi_3dhybrid_bumpcov_bumploc_2pe
```


Generate build files

Compile MPAS-JEDI executables

Test if the code was compiled properly

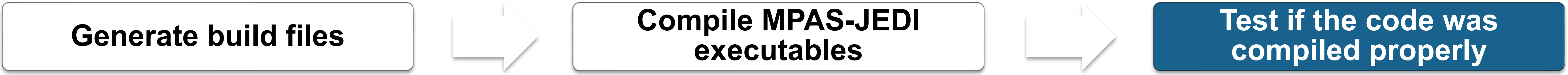
Sample yaml files of ctest cases under ~mpas-jedi/test/testinput/



```
3denvar_2stream_bumploc.yaml   eda_3dhybrid_2.yaml           hofx3d.yaml
3denvar_amsua_allsky.yaml      eda_3dhybrid_3.yaml           hofx.yaml
3denvar_amsua_bc.yaml          eda_3dhybrid_4.yaml           increment.yaml
3denvar_bumploc.yaml           eda_3dhybrid.yaml             letkf_2dloc.yaml
3denvar_dual_resolution.yaml   enshofx_1.yaml                letkf_3dloc.yaml
3dfgat.yaml                    enshofx_2.yaml                lgetkf.yaml
3dhybrid_bumpcov_bumploc.yaml  enshofx_3.yaml                linvarcha.yaml
3dvar_bumpcov_ropp.yaml        enshofx_4.yaml                model.yaml
3dvar_bumpcov_rttovcpp.yaml    enshofx_5.yaml                namelists
3dvar_bumpcov.yaml             enshofx.yaml                  obslocalizations.yaml
3dvar.yaml                     errorcovariance.yaml          obslocalization_vertical.yaml
4denvar_bumploc.yaml           forecast.yaml                 obslocalization.yaml
4denvar_ID.yaml                gen_ens_pert_B.yaml           obsop_name_map.yaml
convertstate_bumpinterp.yaml   geometry_iterator_2d.yaml     parameters_bumpcov.yaml
convertstate_unsinterp.yaml    geometry_iterator_3d.yaml     parameters_bumploc.yaml
dirac_bumpcov.yaml             geometry.yaml                 rtpp.yaml
dirac_bumploc.yaml             getvalues.yaml                state.yaml
dirac_noloc.yaml               hofx3d_ropp.yaml              unsinterp.yaml
eda_3dhybrid_1.yaml            hofx3d_rttovcpp.yaml
```

NCAR | UCAR

JEDI yaml file is like Fortran's namelist



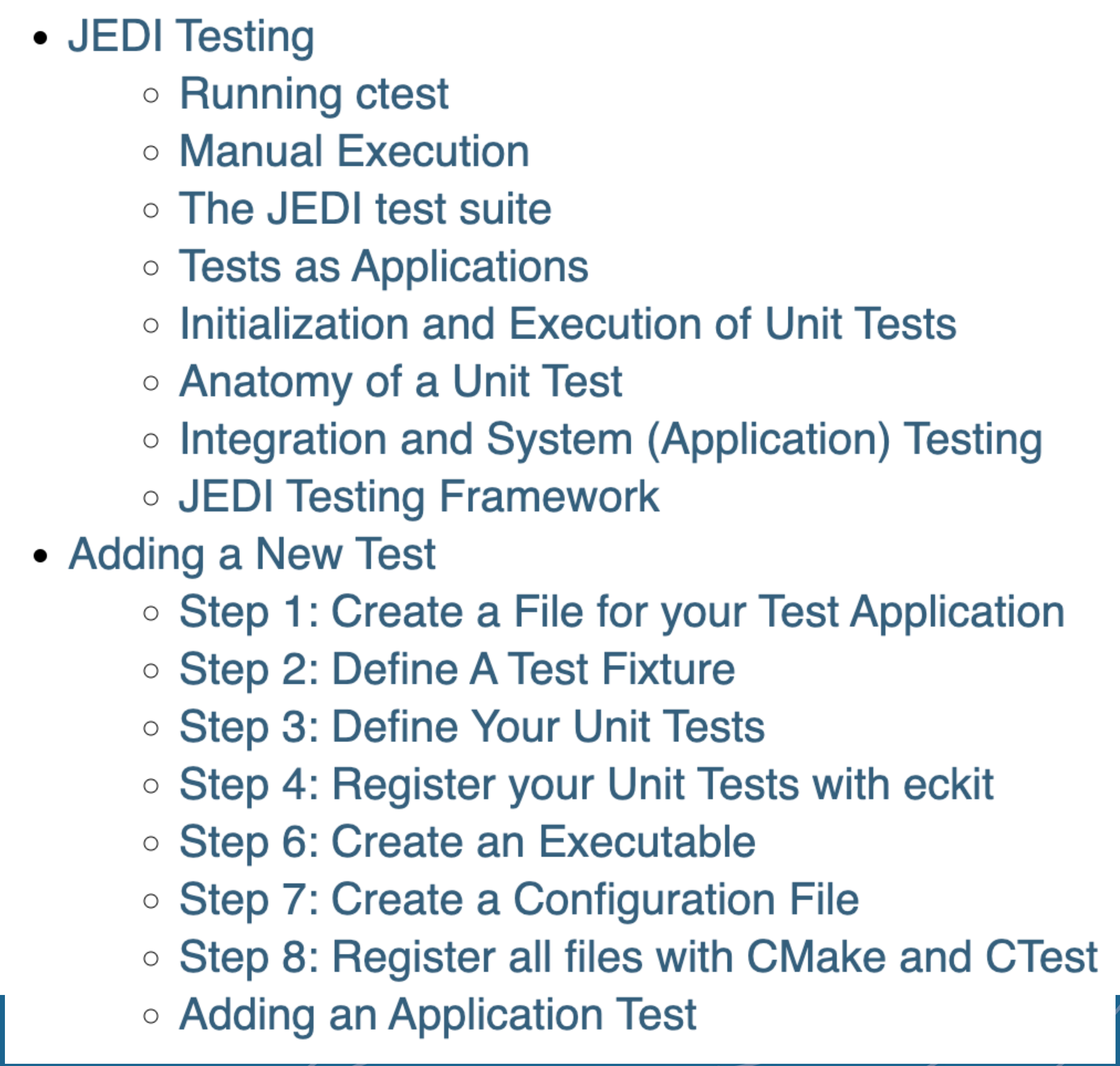
Generate build files

Compile MPAS-JEDI executables

Test if the code was compiled properly

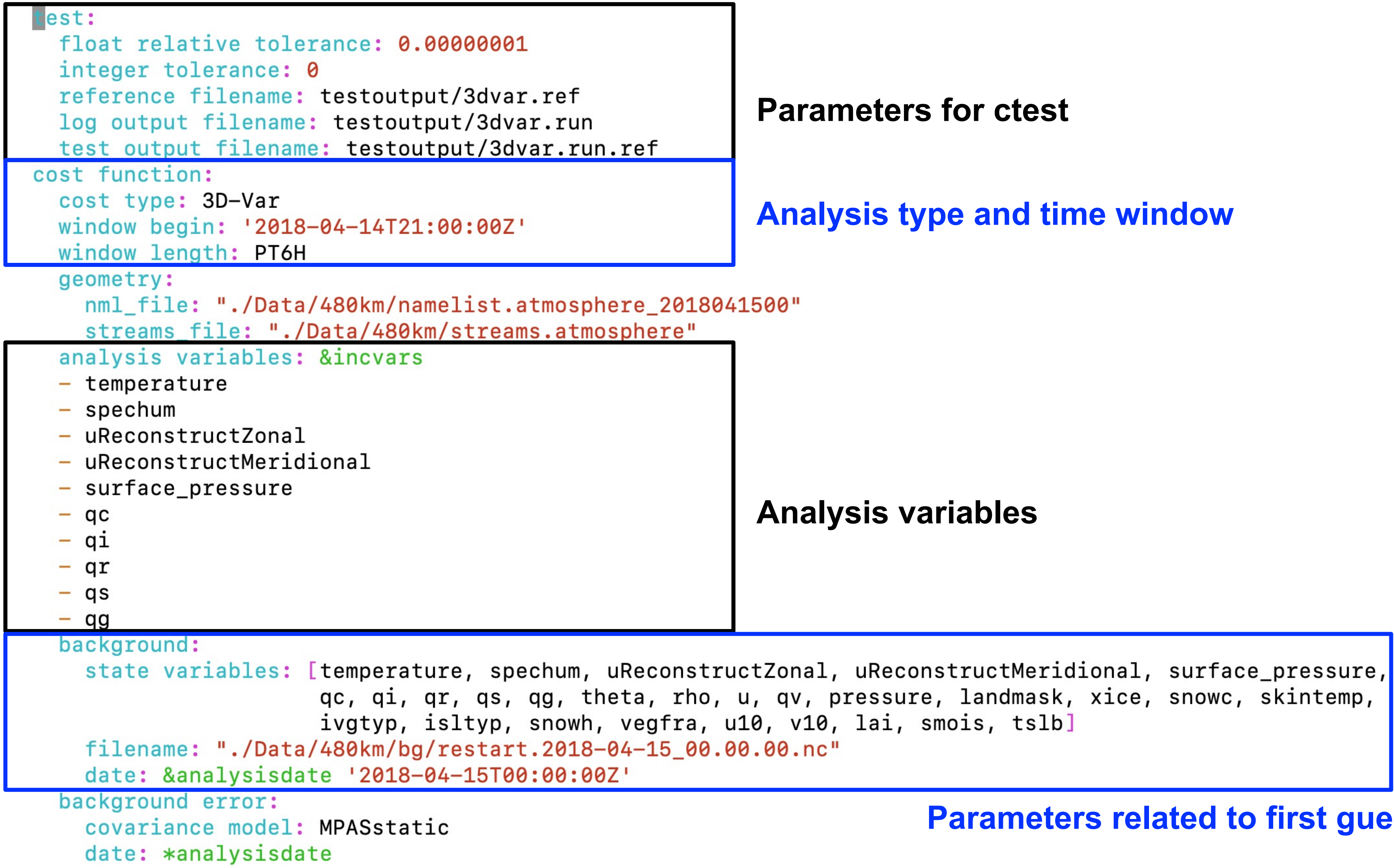
Further reading about JEDI Testing

https://jointcenterforsatellitedataassimilation-jedi-docs.readthedocs-hosted.com/en/latest/inside/testing/index.html



- JEDI Testing
  - Running ctest
  - Manual Execution
  - The JEDI test suite
  - Tests as Applications
  - Initialization and Execution of Unit Tests
  - Anatomy of a Unit Test
  - Integration and System (Application) Testing
  - JEDI Testing Framework
- Adding a New Test
  - Step 1: Create a File for your Test Application
  - Step 2: Define A Test Fixture
  - Step 3: Define Your Unit Tests
  - Step 4: Register your Unit Tests with eckit
  - Step 6: Create an Executable
  - Step 7: Create a Configuration File
  - Step 8: Register all files with CMake and CTest
  - Adding an Application Test

Overview of JEDI yaml file



```yaml
test:
  float relative tolerance: 0.00000001
  integer tolerance: 0
  reference filename: testoutput/3dvar.ref
  log output filename: testoutput/3dvar.run
  test output filename: testoutput/3dvar.run.ref
cost function:
  cost type: 3D-Var
  window begin: '2018-04-14T21:00:00Z'
  window length: PT6H
  geometry:
    nml_file: "./Data/480km/namelist.atmosphere_2018041500"
    streams_file: "./Data/480km/streams.atmosphere"
  analysis variables: &incvars
  - temperature
  - spechum
  - uReconstructZonal
  - uReconstructMeridional
  - surface_pressure
  - qc
  - qi
  - qr
  - qs
  - qg
  background:
    state variables: [temperature, spechum, uReconstructZonal, uReconstructMeridional, surface_pressure,
                      qc, qi, qr, qs, qg, theta, rho, u, qv, pressure, landmask, xice, snowc, skintemp,
                      ivgtyp, isltyp, snowh, vegfra, u10, v10, lai, smois, tslb]
    filename: "./Data/480km/bg/restart.2018-04-15_00.00.00.nc"
    date: &analysisdate '2018-04-15T00:00:00Z'
  background error:
    covariance model: MPASstatic
    date: *analysisdate
```

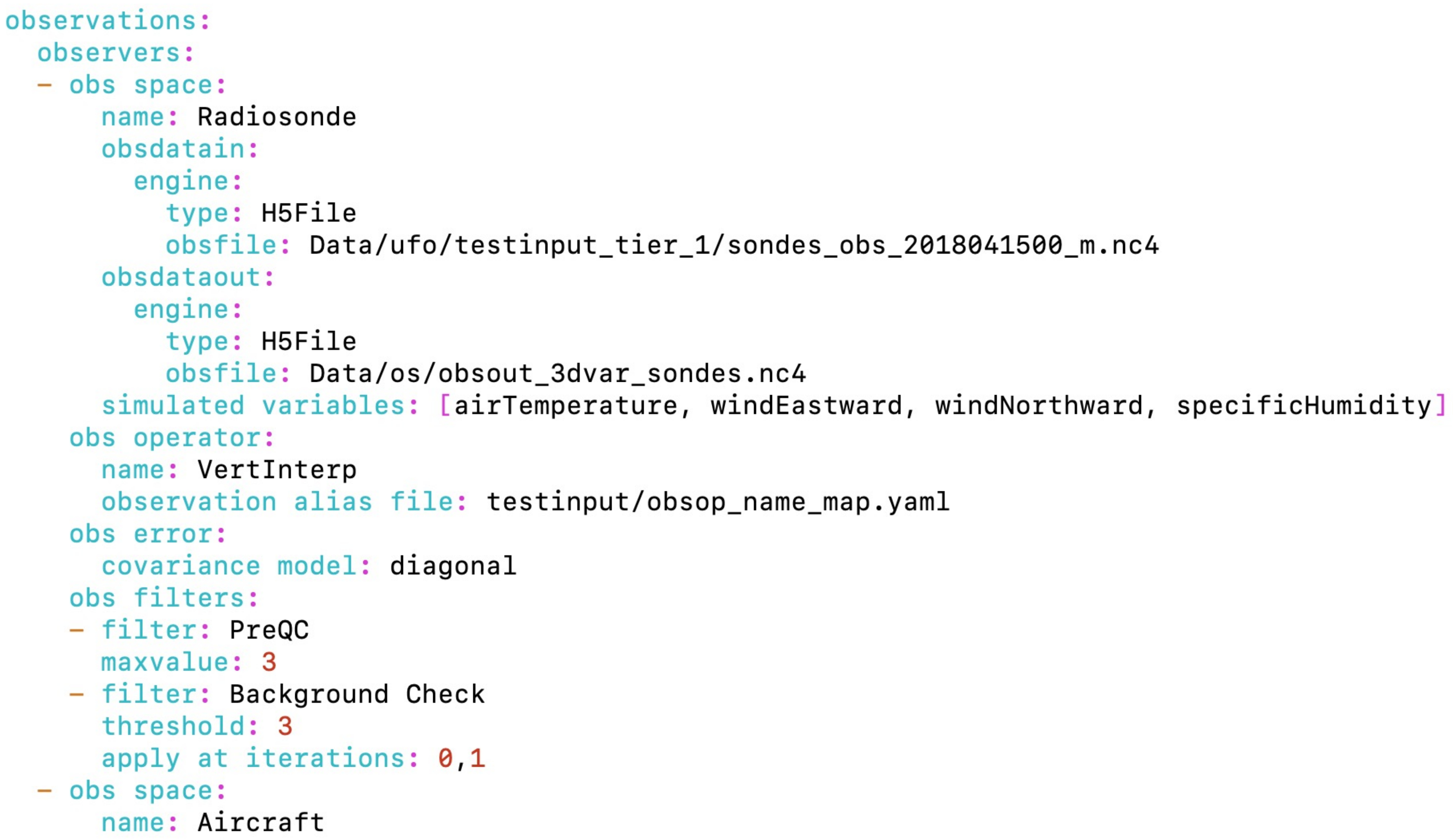
Parameters for ctest

Analysis type and time window

Analysis variables

Parameters related to first guess

Parameters related to observations



```
observations:
  observers:
  - obs space:
      name: Radiosonde
      obsdatain:
        engine:
          type: H5File
          obsfile: Data/ufo/testinput_tier_1/sondes_obs_2018041500_m.nc4
      obsdataout:
        engine:
          type: H5File
          obsfile: Data/os/obsout_3dvar_sondes.nc4
      simulated variables: [airTemperature, windEastward, windNorthward, specificHumidity]
    obs operator:
      name: VertInterp
      observation alias file: testinput/obsop_name_map.yaml
    obs error:
      covariance model: diagonal
    obs filters:
    - filter: PreQC
      maxvalue: 3
    - filter: Background Check
      threshold: 3
      apply at iterations: 0,1
  - obs space:
      name: Aircraft
```

More details on the JEDI YAML file configuration will be provided in the upcoming talks.