This material is based upon work supported by the NSF National Center for Atmospheric Research, which is a major facility operated by the U.S. National Science Foundation under Cooperative Agreement No. 1852977.

Introduction to Practical Sessions

Presented by Ivette Hernández Baños based on materials prepared by I-Han Chen

Prediction, Assimilation, and Risk Communication Section
Mesoscale & Microscale Meteorology Laboratory
National Center for Atmospheric Research

MONAN: INPE MPAS-JEDI Training 2024, Cachoeira Paulista, São Paulo, Brazil
August 15-16, 2024

Instructions for MPAS-JEDI practice exercises

https://www2.mmm.ucar.edu/projects/mpas-jedi/tutorial/202408INPE

EGEON cluster computer system

ssh -X username@egeon-login.cptec.inpe.br

Check what shell you are using
- echo $SHELL

*** use bash**

Submitting Jobs with Slurm

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

It is recommended that all code be **built** on the "**egeon-login**.cptec.inpe.br" host rather than a **head node**

the **"head node" can be very slow!!!*

Obtain the mpas_jedi_tutorial folder

cd /mnt/beegfs/${USER}

cp -r /mnt/beegfs/professor/mpasjedi_tutorial/mpasjedi_tutorial2024_testdata ./mpas_jedi_tutorial

(this will create your own working directory that contains all data needed for the tutorial: /mnt/beegfs/${USER}/mpas_jedi_tutorial)

ls -l mpas_jedi_tutorial, *you will see:*

```
total 7
drwxr-sr-x 3 professor professor   1 Aug  7 19:40 background
drwxr-sr-x 3 professor professor   1 Aug  7 19:36 background_120km
drwxr-sr-x 5 professor professor   3 Aug  7 19:39 B_Matrix
drwxr-sr-x 4 professor professor  38 Aug  9 06:18 conus15km
drwxr-sr-x 2 professor professor  27 Aug  7 19:36 crtm_coeffs_v3
drwxr-sr-x 3 professor professor   1 Aug  7 19:36 ensemble
drwxr-sr-x 2 professor professor 168 Aug  7 19:36 localization_pregenerated
drwxr-sr-x 2 professor professor  35 Aug 10 00:59 MPAS_JEDI_yamls_scripts
drwxr-sr-x 2 professor professor  30 Aug  8 04:13 MPAS_namelist_stream_physics_files
drwxr-sr-x 2 professor professor   6 Aug  7 19:40 ncl_scripts
drwxrwsr-x 5 professor professor   7 Aug  9 21:58 obs2ioda_prebuild
drwxr-sr-x 3 professor professor   7 Aug  8 19:11 obs_bufr
drwxr-sr-x 3 professor professor   1 Aug  7 19:35 obs_ioda_pregenerated
drwxr-sr-x 4 professor professor   2 Aug  9 00:41 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 EGEON

- **source ../code/env-setup/gnu-<machine>.sh**
- **module list (100 modules!!!!!)**

```
Currently Loaded Modules:
  1) ecflow/5.8.4                 35) base-env/1.0.0          69) json-schema-validator/2.1.0
  2) mysql/8.0.33                 36) boost/1.83.0            70) odc/1.4.6
  3) ncarenv/23.09          (S)   37) openblas/0.3.24         71) py-attrs/21.4.0
  4) gcc/12.2.0                   38) py-setuptools/63.4.3    72) py-pycparser/2.21
  5) stack-gcc/12.2.0             39) py-numpy/1.22.3         73) py-cffi/1.15.1
  6) craype/2.7.20                40) bufr/12.0.1             74) py-findlibs/0.0.2
  7) cray-mpich/8.1.25            41) ecbuild/3.7.2           75) py-eccodes/1.5.0
  8) libfabric/1.15.2.0           42) libpng/1.6.37           76) py-f90nml/1.4.3
  9) cray-pals/1.2.11             43) openjpeg/2.3.1          77) py-h5py/3.7.0
 10) stack-cray-mpich/8.1.25      44) eccodes/2.32.0          78) py-cftime/1.0.3.4
 11) tar/1.34                     45) eigen/3.4.0             79) py-netcdf4/1.5.8
 12) gettext/0.21.1               46) eckit/1.24.5            80) py-bottleneck/1.3.7
 13) libxcrypt/4.4.35             47) fftw/3.3.10             81) py-numexpr/2.8.4
 14) zlib/1.2.13                  48) fckit/0.11.0            82) py-et-xmlfile/1.0.1
 15) sqlite/3.43.2                49) fiat/1.2.0              83) py-openpyxl/3.1.2
 16) util-linux-uuid/2.38.1       50) ectrans/1.2.0           84) py-six/1.16.0
 17) python/3.10.13               51) qhull/2020.2            85) py-python-dateutil/2.8.2
 18) stack-python/3.10.13         52) atlas/0.35.1            86) py-pytz/2023.3
 19) nghttp2/1.57.0               53) git-lfs/3.3.0           87) py-pyxlsb/1.0.10
 20) curl/8.4.0                   54) gsibec/1.1.3            88) py-xlrd/2.0.1
 21) cmake/3.23.1                 55) gsl-lite/0.37.0         89) py-xlsxwriter/3.1.7
 22) git/2.41.0                   56) libjpeg/2.1.0           90) py-xlwt/1.3.0
 23) pkg-config/0.29.2            57) krb5/1.19.2             91) py-pandas/1.5.3
 24) hdf5/1.14.0                  58) libtirpc/1.3.3          92) py-pybind11/2.11.0
 25) snappy/1.1.10                59) hdf/4.2.15              93) py-pycodestyle/2.11.0
 26) zstd/1.5.2                   60) jedi-cmake/1.4.0        94) py-pyhdf/0.10.4
 27) c-blosc/1.21.5               61) libxt/1.1.5             95) libyaml/0.2.5
 28) netcdf-c/4.9.2               62) libxmu/1.1.4            96) py-pyyaml/6.0
 29) nccmp/1.9.0.1                63) libxpm/3.5.12           97) py-scipy/1.11.3
 30) netcdf-fortran/4.6.1         64) libxaw/1.0.13           98) py-packaging/23.1
 31) parallel-netcdf/1.12.2       65) udunits/2.2.28          99) py-xarray/2023.7.0
 32) parallelio/2.5.10            66) ncview/2.1.9           100) sp/2.5.0
 33) py-pip/23.1.2                67) netcdf-cxx4/4.3.1      101) jedi-base-env/1.0.0
 34) wget/1.20.3                  68) json/3.10.5            102) jedi-mpas-env/1.0.0

  Where:
   S:  Module is Sticky, requires --force to unload or purge
```

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

Required spack-stack build environment

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

- **source ../code/env-setup/gnu-egeon.sh**
- **module list (here reduced to 31 modules!!!)**

```
Currently Loaded Modules:
  1) stack-gcc/12.2.0      7) libfabric/1.21.0  13) gsl-lite/0.37.0         19) parallelio/2.6.2
  2) binutils/2.42         8) atlas/0.36.0      14) hdf5/1.14.3             20) boost/1.84.0
  3) gcc/12.2.0            9) eckit/1.24.5      15) netcdf-c/4.9.2          21) cmake/3.27.9
  4) openssh/9.7p1        10) fckit/0.11.0      16) netcdf-cxx4/4.3.1       22) jedi-cmake/1.4.0
  5) openmpi/5.0.3        11) fftw/3.3.10       17) netcdf-fortran/4.6.1    23) ecbuild/3.7.2
  6) stack-openmpi/5.0.3  12) gptl/8.1.1        18) parallel-netcdf/1.12.3  24) eigen/3.4.0

 25) python-venv/1.0        31) nccmp/1.9.0.1
 26) py-setuptools/63.4.3
 27) py-pycodestyle/2.11.0
 28) sqlite/3.43.2
 29) openblas/0.3.24
 30) udunits/2.2.28
```

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

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

Clone mpas-bundle repository and checkout the ‘release/3.0.0-beta’ branch

- **cd /mnt/beegfs/${USER}/mpas_jedi_tutorial**
- **mkdir mpas_bundle_v3**
- **cd mpas_bundle_v3**
- **git clone -b release/3.0.0-beta https://github.com/liujake/mpas-bundle code**

```
Cloning into 'code'...
remote: Enumerating objects: 485, done.
remote: Counting objects: 100% (106/106), done.
remote: Compressing objects: 100% (60/60), done.
remote: Total 485 (delta 60), reused 86 (delta 46), pack-reused 379
Receiving objects: 100% (485/485), 155.47 KiB | 1.49 MiB/s, done.
Resolving deltas: 100% (289/289), done.
```

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

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

- **vi code/CMakeLists.txt**

JEDI components

Repositories and branch/tag information

```
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 )
```

default build option

Note 1: 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”.

Note 2: We will use the pre-build single-precision mpas-bundle executable in practical sessions

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

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*

2. Compile MPAS-JEDI executables (make)

- **make -j14**

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

(The compilation will take ~14 min to complete)

MPAS-JEDI related executables under ~build/bin

```
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
```

How to obtain, compile, and test the MPAS-JEDI for this tutorial?

We will use a **'local'** version of the source code and test dataset:

cd /mnt/beegfs/${USER}/mpas_jedi_tutorial

cp /mnt/beegfs/professor/mpasjedi_tutorial/mpasbundlev3_local.tar.gz .

tar xzf mpasbundlev3_local.tar.gz; ls -l mpasbundlev3_local/*

```
mpasbundlev3_local/build:
total 1
drwxrwsr-x 3 professor professor 1 Aug 13 21:15 test_data

mpasbundlev3_local/code:
total 34
-rw-rw-r--  1 professor professor  5032 Aug 13 21:28 CMakeLists.txt
drwxrwsr-x 11 professor professor    21 Aug 13 21:08 crtm
drwxrwsr-x  2 professor professor     7 Aug 14 08:58 env-setup
drwxrwsr-x 10 professor professor    18 Aug 13 21:09 ioda
drwxrwsr-x  5 professor professor     5 Aug 13 21:12 ioda-data
-rw-rw-r--  1 professor professor 11334 Aug 13 21:09 LICENSE
drwxrwsr-x  8 professor professor    12 Aug 13 21:09 MPAS
drwxrwsr-x 11 professor professor    17 Aug 13 21:09 mpas-jedi
drwxrwsr-x  4 professor professor     8 Aug 13 23:35 mpas-jedi-data
drwxrwsr-x 13 professor professor    23 Aug 13 21:08 oops
-rw-rw-r--  1 professor professor 10146 Aug 13 21:09 README.md
drwxrwsr-x 12 professor professor    22 Aug 13 21:12 saber
drwxrwsr-x  2 professor professor     2 Aug 13 21:10 scripts
drwxrwsr-x  3 professor professor     2 Aug 13 21:12 test-data-release
drwxrwsr-x 11 professor professor    19 Aug 13 21:10 ufo
drwxrwsr-x  6 professor professor     7 Aug 13 23:58 ufo-data
drwxrwsr-x  9 professor professor    14 Aug 13 21:12 vader
```

How to obtain, compile, and test the MPAS-JEDI for this tutorial?

cd mpasbundlev3_local/build

source ../code/env-setup/gnu-egeon.sh

module list

```
Currently Loaded Modules:
  1) stack-gcc/12.2.0      7) libfabric/1.21.0  13) gsl-lite/0.37.0         19) parallelio/2.6.2
  2) binutils/2.42         8) atlas/0.36.0      14) hdf5/1.14.3             20) boost/1.84.0
  3) gcc/12.2.0            9) eckit/1.24.5      15) netcdf-c/4.9.2          21) cmake/3.27.9
  4) openssh/9.7p1        10) fckit/0.11.0      16) netcdf-cxx4/4.3.1       22) jedi-cmake/1.4.0
  5) openmpi/5.0.3        11) fftw/3.3.10       17) netcdf-fortran/4.6.1    23) ecbuild/3.7.2
  6) stack-openmpi/5.0.3  12) gptl/8.1.1        18) parallel-netcdf/1.12.3  24) eigen/3.4.0

 25) python-venv/1.0        31) nccmp/1.9.0.1
 26) py-setuptools/63.4.3
 27) py-pycodestyle/2.11.0
 28) sqlite/3.43.2
 29) openblas/0.3.24
 30) udunits/2.2.28
```

How to obtain, compile, and test the MPAS-JEDI for this tutorial?

CMakeLists.txt was modified for this tutorial so that cmake will **NOT** fetch code and test data from github, instead, it uses the pre-downloaded local **code/testdata** under "**../code**".

"Makefile" files to build executables are generated under the **build** directory

```
make -j18
```

```
$ ls bin/mpas*
bin/mpas_atmosphere
bin/mpas_atmosphere_build_tables
bin/mpas_init_atmosphere
bin/mpasjedi_convertstate.x
bin/mpasjedi_converttostructuredgrid.x
bin/mpasjedi_eda.x
bin/mpasjedi_enkf.x
bin/mpasjedi_enshofx.x
bin/mpasjedi_ens_mean_variance.x
bin/mpasjedi_error_covariance_toolbox.x
bin/mpasjedi_forecast.x
bin/mpasjedi_gen_ens_pert_B.x
bin/mpasjedi_hofx3d.x
bin/mpasjedi_hofx.x
bin/mpasjedi_rtpp.x
bin/mpasjedi_saca.x
bin/mpasjedi_variational.x
bin/mpas_namelist_gen
bin/mpas_parse_atmosphere
bin/mpas_parse_init_atmosphere
bin/mpas_streams_gen
```

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

export **LD_LIBRARY_PATH**=/mnt/beegfs/${USER}/mpas_jedi_tutorial/mpasbundlev3_local/build/lib:$LD_LIBRARY_PATH

cd mpas-jedi

ctest

```
......
82% tests passed, 10 tests failed out of 57

Label Time Summary:
executable    =  24.35 sec*proc (13 tests)
mpasjedi      = 265.92 sec*proc (57 tests)
mpi           = 261.83 sec*proc (56 tests)
script        = 241.57 sec*proc (44 tests)

Total Test time (real) = 266.22 sec

The following tests FAILED:
          8 - test_mpasjedi_unsinterp_4pe (Failed)
          9 - test_mpasjedi_geometry_iterator_2d_2pe (Failed)
         10 - test_mpasjedi_geometry_iterator_3d_2pe (Failed)
         50 - test_mpasjedi_letkf_3dloc_4pe (Failed)
         51 - test_mpasjedi_lgetkf_4pe (Failed)
         53 - test_mpasjedi_forecast_2pe (Failed)
         54 - test_mpasjedi_parameters_bumpcov_2pe (Failed)
         55 - test_mpasjedi_parameters_bumploc_2pe (Failed)
         56 - test_mpasjedi_3dvar_2pe (Failed)
         57 - test_mpasjedi_3dhybrid_bumpcov_bumploc_2pe (Failed)
```

Ideally: all 57 ctest cases pass!!!

However, 10 cases of using more than 1 core are failing on Egeon for some unknown reason. This will not affect our tutorial test cases.

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

Use ctest to ensure that the code was compiled properly

- **cd mpas-jedi**
- **ctest**

(takes ~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
```

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

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      → existing reference file
4denvar_ID.run      → full text log file for the present test
4denvar_ID.run.ref  → shortened reference file
convertstate_bumpi    (part of the 4denvar_ID.run)
convertstate_bumpinterp.run
convertstate_bumpinterp.run.ref
convertstate_unsinterp.ref
```

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

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

‘ctest –N’ will list, but not run 57 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
```

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

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

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

JEDI yaml file is like Fortran's namelist

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

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

Example of a JEDI yaml file

```
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/background

Parameters related to observations

```yaml
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