5.6 Acceptance testing OpenIFS

Prerequisites

You should have completed:

- Installation of data files
- Successful compilation of OpenIFS.

Introduction

The OpenIFS tarfile distribution includes a simple low resolution (T21) test job that can be used to verify the model is working correctly in the directory: `oifs/t21test`. It is **strongly** recommended that after compiling the model successfully, these short tests are run to verify the model before any development or production is started. These short tests are also a good way to become familiar with the model.

- Single task / single thread.
- 2 x OpenMP threads
- 2 x MPI tasks
- 2 x threads + 2 tasks.
- **Acceptance** (reference) tests. It's strongly recommended this test is completed before using the model for any real use.

These are described below.

OpenIFS 43r3+ also includes a t21test_xios directory, used in a similar way to t21test but including the XIOS parallel I/O library. In order to use XIOS, OpenIFS must be configured and built with the XIOS library. See 5.5 Building OpenIFS for more details. For more information on using XIOS with OpenIFS, please see OpenIFS: Compiling and installing XIOS.

Test directory

The directory `oifs/t21test` contains a number of files:

```
% cd oifs
% ls t21test
ICMGGepc8INIT  ICMGGepc8INIUA  ICMSHepc8INIT  README_t21test  fort.4
ifsdata  job.sh  ref_021_0072
```

Files beginning with 'ICM'.
These are the input files for this T21 experiment. They are in GRIB format. Do not move them from this directory. OpenIFS expects to find it's input files in the same directory as the main executable.

epc8 - this is the Experiment ID.
ICMGGepc8 - 'GG' indicates these contain gridpoint fields. There are two types of 'GG' files: ICMGGepc8INIUA are the 'Upper Air' gridpoint fields; ICMGGepc8INIT are single level 'surface' fields only.
ICMSHepc8 - 'SH' indicates these contains spherical harmonic fields.

You can use the '`grib_ls`' and '`grib_dump`' commands to see the contents of these files (the commands will be in the 'bin' directory of your eccodes installation).

job.sh
Simple shell script to run the model. Described in more detail below. Type '`job.sh -h`' to get usage information.

ifsdata
Climate data fields used for T21 test integration. You should not move or rename this directory as the model will expect to find the climate files it needs in a directory of this name.

fort.4
This file contains all of the input model fortran NAMELISTS. Not all of the namelists have their variables listed, only the variables commonly changed are listed here.

On this page ...

In this section...

OpenIFS User Guide ...

ref_021_0072
This file is reference output for the model tests. The model can be run in 'reference' mode where it checks it is working correctly by comparing some mathematical norms against these files. Reference runs are described in more detail under 'Acceptance testing' below.

Test integrations

A number of **short model runs are strongly recommended to verify the model** is working correctly. Once you have compiled the model without errors, follow these steps.

These tests will ensure the model can run with multiple OpenMP threads, with MPI tasks and in mixed OpenMP/MPI mode. A further acceptance test can be run which compares the model output on your machine with reference data obtained from machines at ECMWF.

Serial : single task, single thread.

1. **Build the model** if you haven't already, in the `oifs/make` directory. Use the optimized ('opt') build configuration. If you find these tests do not work with this configuration, and you have not modified the compilation options, report the problem to openifs-support@ecmwf.int. In the meantime, try using the 'noopt' configuration and then experiment by raising the optimization level.

2. **Copy the OpenIFS executable, master.exe**, from the make directory e.g. '`make/gnu-opt/oifs/bin/master.exe`' (or `make/intel-opt/oifs/bin/master.exe`) to the `t21test` directory. Although the path to the model executable can be given on the command line, it's better for these tests to have a copy in the `t21test` directory.

3. Run the model with a **single task and single thread** by executing the job script:

```
% ./job.sh -e epc8 -n 1 -t 1 -x ./master.exe
```

The model will expect to find the namelist file called `fort.4` in the same directory as the executable.

If the run works you will see output like:

```
...
signal_drhook(SIGSYS=31): New handler installed at 0x4d06cf; old preserved
at 0x0
MPL_BUFFER_METHOD:  2           0
   16:03:46 STEP    0 H=   0:00 +CPU=  3.598
   16:03:46 STEP    1 H=   0:10 +CPU=  0.535
   16:03:47 STEP    2 H=   0:20 +CPU=  0.537
   16:03:48 STEP    3 H=   0:30 +CPU=  0.537
   16:03:48 STEP    4 H=   0:40 +CPU=  0.527
   16:03:49 STEP    5 H=   0:50 +CPU=  0.526
   16:03:49 STEP    6 H=   1:00 +CPU=  0.530
```

This test runs only 6 timesteps.

If the job **command can't find the model executable,** make sure you have copied the `master.exe` file from the 'make/*/oifs/bin' directory to the t21test directory.

OpenIFS output

OpenIFS writes its output to the same directory as the executable. Output consists of several filetypes. The job.sh script moves these files on successful completed to the '**output0**' directory.

NODE_001.01 contains the text output (WRITE/PRINT statements). The numbers refer to task number and thread number. Only output from the master task & thread is normally output but this can be changed for debugging purposes.

ICM*epc8+0000 is the model output in GRIB format split into two types of files; one for the gridpoint, the other for spectral fields. These contain only a few output variables in this test. This file is a mix of GRIB1 and GRIB2 messages.

ifs.stat is a small file that prints the model steps, time taken for each step and a 'norm' measure. This file can be usually ignored but is useful for debugging.

Note that OpenIFS will **appends** to any existing output GRIB (i.e. ICM*+*) files in the same directory as the executable, rather than overwriting them.

Possible errors

Missing GRIB samples

The model will fail with an error if it cannot find the '`ifs_samples`' directory in the eccodes installation:

```
signal_drhook(SIGSYS=31): New handler installed at 0x4d06cf; old preserved
at 0x0
MPL_BUFFER_METHOD:  2           0
ECCODES ERROR   :  Unable to locate sample file gg_sfc_grib1.tmpl
                    in /home/auser/ecmwf/eccodes/ifs_samples/grib1_mlgrib2
 GRIB_NEW_FROM_TEMPLATE gg_sfc_grib1 FAILED          -2
```

Fix: Check the location of the grib samples directory and set the `GRIB_IFS_SAMPLES_PATH` environment variable in the `oifs-config.sh` file (see 5.5 Building OpenIFS), to give the location of the file '`grib1_mlgrib2`' in your eccodes installation, in the directory '`ifs_samples`'.

This error can often arise if the eccodes directory has been moved since compilation as the ifs_samples path is hardcoded at compile time.

Memory fault / Segmentation fault

If you see an error like this:

```
mpirun noticed that job rank 0 with PID 7429 on node elvira exited on
signal 11 (Segmentation fault).
```

or

```
MEMORY FAULT
```

it is most likely OpenIFS requires more 'stack' memory than your default setting. This can happen when increasing the number of OpenMP threads.

To solve the problem add the line:

```
ulimit -s unlimited
```

to the file 'job' just before the oifs_run line. This will increase the per-process stack memory limit to the maximum the operating system allows.

If the model still fails, contact openifs-support@ecmwf.int for assistance.

Parallel: 2 threads and 2 tasks

These next short tests verify the model works correctly with either OpenMP parallel threading, MPI tasks and both, and follow on from the serial test above.

OpenMP threads is only enabled for optimized 'opt' builds

(a) Two OpenIFS threads:

Increase the number of threads on the run command:

```
% ./job.sh -e epc8 -n 1 -t 2 -x ./master.exe
```

If this works, look in the NODE_001.01 output file (in directory output0) for the line:

```
NUMBER OF THREADS                2
```

to verify the model ran with 2 OpenMP threads.

b) Two MPI tasks:

Edit the fort.4 file and change `NPROC` to 2. Note that increasing the number of tasks requires changing the number of tasks in both the model namelist and the command line.

Rerun the job:

```
% ./job.sh -e epc8 -n 2 -t 1 -x ./master.exe
```

and again look in the NODE_001.01 output file for the line: "`NUMBER OF TASKS      2`" to verify that two MPI tasks was used.

Mixed mode: OpenMP and MPI

If the short tests above succeed, rerun again with both MPI tasks and OpenMP threads increased:

```
% ./job.sh -e epc8 -n 2 -t 2 -x ./master.exe
```

confirm that 2 threads and 2 tasks was used in the NODE file and the run was successful.

You should also notice the CPU time reported as the model runs is now less than with 1 task and 1 thread.

If all these technical tests work, perform the acceptance test below to ensure that the numerical results from the model are as expected. It's **strongly recommended** this test is completed before proceeding to work with the model for development or production.

Acceptance testing

The final step is to check the model is producing the numerical answers within acceptable limits, even if it runs the short tests above without failing.

OpenIFS includes code that will compute internal statistical norms and compare against numbers supplied by ECMWF. The file: `ref_021_0072` in the `t21test` directory contains statistical norms computed by the model run at ECMWF.

> OpenIFS CY38 releases used a longer run of 144 steps (24hrs). In later releases this was changed to 12hrs.

Before running the test, change the number of MPI tasks and OpenMP threads back to 1. It is prudent to run the test without any parallel execution because experience shows that some compiler libraries have internal threading which can cause differences in the results.

Remember to set `NPROC=1` in the fort.4 namelist file as well as on the job.sh command line (see above).

Namelist changes

To do the acceptance test, edit the namelists in `fort.4` and look for the **NAMCT0** namelist:

```
&NAMCT0
 LREFOUT=false,
 NSTOP=6,
```

change the number of timesteps to 72 to run the model for 12hrs (assuming you have not changed the default timestep of 10mins at T21) and set the **LREFOUT to TRUE**:

```
&NAMCT0
 LREFOUT=true,
 NSTOP=72,
```

With LREFOUT=true, at the last timestep OpenIFS will read the ref_021_0072 file and produce a new file: res_021_0072 (note the similar filenames!). The contents of the file should be similar to:

```
% cat res_021_0072

               Results of ERROR calculation

 The error calculated from the results shows
 that the calculations are correct

 The maximum error is =         0.11345 %
```

The maximum error should be below 2-3%. The value of 0.11345 is illustrative in this example.

As long as the model reports 'calculations are correct' the model is behaving satisfactorily in your compilation and run environment.

However, note that the ref_021_0072 file was generated by using the GNU compilers. If you use a different compiler such as Intel, you will see a larger maximum error value.

Generating validation tests

To generate additional validation tests to produce your own ref* files, use the namelist switch:

NAMCT0
LREFGEN=.true.,

With this set, run the model for a short forecast. At the end of the run, a ref_*_* file will be created with the resolution value and the total number of steps in the filename.

The model should not be run for long because this test relies on a linear growth of errors. A 12 hour run is generally recommended, particularly at higher resolutions.

Any questions/problems please contact openifs-support@ecmwf.int.