Tutorial: GLORI4DE scenarios

16/05/2025

1. Introduction

This tutorial provides guidance for running the four GLORI4DE scenarios implemented through ecFlow suites. The scenarios represent different configurations of the ICON model workflow for weather prediction and data assimilation. Two floods over the Emilia Romagna region of Italy will be used as examples throughout the tutorial:

- the first one on 16 and 17/05/2023,
- the second one on 19/10/2024.

The code provided and installation guides have been designed and tested on the HPC site "Leonardo", from CINECA.

Refer to the README.md in the Github repository, as well as the individual "readme" and "how to" files associated with each tool or scenario, for:

- Repository structure,
- System requirements,
- Software dependencies.

2. Preparation

2.1 Software installation

Ensure these components are installed following their respective documentation:

- **BACY** (workflow manager developed by DWD)
  More info at: https://gitlab.dkrz.de/dace/dace_bacy.git
- **ecFlow** (workflow manager developed by ECMWF)
  More info in: `tools/ecflow/how_to_install_ecflow_on_leonardo.md`
- **nwprun** (workflow manager based on ecFlow and designed by ARPAE-SIMC)
  More info in: `scenario_3/README.txt`
- **DACE** (data assimilation components developed by DWD)
  More info in: `tools/dace/how_to_install_DACE.md`

2.2 Configuration files

Replace default configuration files and job templates with those provided in this repository.

For BACY, the files are located in the sub-directories associated with the individual modules, located inside the following directories:

- `scenario_1/global/bacy/modules`
- `scenario_1/lam/bacy/modules`

- `scenario_2_da/lam/bacy/modules`
- `scenario_2_s3/lam/bacy/modules`
- `scenario_3/bacy/lam/bacy_remap/modules`
- `scenario_3/bacy/lam/bacy_more/modules`
- `scenario_4_da/global/bacy/modules`
- `scenario_4_s3/global/bacy/modules`

For nwprun, the files are located within the sub-directories of:

- `scenario_3/nwprun/conf`
- `scenario_3/nwprun/ecflow`

2.3 Symbolic links setup

In each scenario's “`bacy_data/bin`” directory, create these required symbolic links:

- `icon` executable and `wrapper` for the current HPC site,
- `iconremap` from icontools,
- `cdo` executable, whose version depends on the module loaded,
- `bufr2netcdf` executable, which on Leonardo is located at `/leonardo_work/smr_prod/lami/smnd-2.7/bin/bufr2netcdf` ,
- `dbamsg` executable, which on Leonardo is located at `/leonardo_work/smr_prod/lami/smnd-2.7/bin/dbamsg` ,
- `var3d` executable from the DACE software, needed for data assimilation.

It is also recommended to link “`ecflow_client`” inside “`ecflow`” directory of each scenario. For scenario 3, the latter is located inside the “`bacy`” sub-directory (`scenario_3/bacy/ecflow`).

2.4 ecFlow server setup

Start the ecFlow server and note the port number:

```
ecflow_server &
```

If the installation instruction provided in this directory have been followed, this executable will be located inside the directory:

```
tools/ecflow/bin
```

It is also possible to specify the port number when starting the server:

```
ecflow_server --port=31415 &
```

The port number needs to be specified inside the `.ecf_passwd` files, located at the home path of the user running the scenario.

The executable that starts the UI is located in the same directory, and it can be started with:

```
ecflow_ui  &
```

3. Scenarios overview

This section provides a brief overview of the scenarios and their individual components.

3.1 Initialization sources

Each scenario requires initial conditions (IC) and, in the case of Limited Area Model (LAM) runs, boundary conditions (BC) from one of two sources:

- IFS, provided by ECMWF and directly available on Leonardo,
- ICON, provided by DWD and transferred via Amazon S3.

3.2 Workflow managers

The choice of workflow manager depends on the IC/BC source:

- For ICON-derived IC and BC:
  - The BACY workflow manager performs data assimilation, remapping and model runs.
  - BACY comprises specialized modules (e.g., for data assimilation, remapping).
  - The cycle module acts as a wrapper, performing complex tasks by calling other modules.
- For IFS-derived IC and BC:
  - nwprun handles data assimilation and IC remapping.
  - BACY manages BC remapping and model execution.

Each scenario implementation has an associated ecFlow suite, generated in one of two ways:

- Directly by nwprun,
- Via a Python script, which calls one of two options:
  - the BACY cycle module, which performs all tasks via internal calls to the modules,
  - individual BACY modules, which allows a more granular control of the tasks.

3.2 Scenario order in the tutorial

This tutorial presents scenarios in increasing order of complexity:

- **Scenario 4**
  Steps:
  - Download and place IC (from ICON),
  - Execute a global ICON run via a single ecFlow task calling the BACY cycle.
- **Scenario 2**
  Steps:
  - Download and place IC/BC (from ICON),
  - Perform a LAM run using multiple ecFlow tasks that call individual BACY modules.
- **Scenario 1**
  Combines elements of Scenarios 4 and 2:
  - Download and place IC (from ICON),
  - Global run (BACY cycle),
  - LAM run (individual BACY modules).
- **Scenario 3**
  Steps:
  - Uses IFS for IC/BC, relying on nwprun for data assimilation.
  - BC remapping and model execution are handled by BACY modules, each called by dedicated ecFlow tasks.

4. Scenario 4

This scenario has two implementation:

1. One uses the data transferred by Amazon S3 from the Horeka HPC site, at the Karlsruhe Institute of Technology (KIT). In this implementation, the analysis is provided by DWD and we do not perform data assimilation. Instead, we execute directly the model run by calling the BACY cycle.
2. The second one uses pre-downloaded data, already stored on Leonardo, and performs the data assimilation using BACY. The observations and initial conditions that we used for testing this implementation have been provided by DWD.

Running both versions of the scenario requires the execution of the same set of steps:

1. Creating the ecFlow suite,
2. Uploading and starting it on the `ecflow_server`,
3. Monitoring the execution via `ecflow_ui`.

4.1 Scenario structure

The parent directory of the scenario (`scenario_4_da` or `scenario_4_s3`) contains the following sub-directories:

- `bacy_data` , which stores the inputs for BACY (IC/BC, grids, observations).
  In this repository, we included the following empty sub-directories:
  - `data`, where the initial conditions are stored.
    In the case of `scenario_4_s3`, it is populated automatically by the ecFlow suite.
  - `routfoxfox`, which contains the grids. The exact path used is specified by the variable `BA_ICON_GRIDDIR` inside the `bacy_conf.sh` script, stored inside the directory `scenario_4_[s3` or `da]/global/bacy/modules/common`.
  - `obs_global`, where the observations are stored,
    It is used only `scenario_4_da` for data assimilation.

- `ecflow_suite`, which contains the Python script that generates the suite, together with the necessary templates and auxiliary scripts.
  The content included in this repository is:
  - `ecf_files`, which is the directory that contains the definition scripts, header, and tail.
  - `scripts`, which contains the script used to download the data from the S3 bucked provided by DWD. It is only included in `scenario_4_s3`.
  - `define_suite_scenario_4.py`, which creates the ecFlow suite.

- `global`, which contains the current instance of BACY.
  For this scenario, it contains one sub-directory:
  - `bacy`, populated by cloning BACY (https://gitlab.dkrz.de/dace/dace_bacy.git).
    In this repository, we included only the configuration files and templates needed by the following modules:
    - `modules/common`, in which the general configuration are specified.
    - `modules/core`, used only by `scenario_4_da` to perform the data assimilation.
    - `modules/more`, which performs the model run.
    - `modules/cycle`, which calls the previous module to perform all needed tasks.

4.2 Creation of the ecFlow suite

Before creating the suite, activate the Python environment with ecFlow:
`source tools/venv_ecflow/bin/activate`

In this repository, we included the empty directory `tools/venv_ecflow` . It corresponds to the relative location in which the virtual environment has been installed on Leonardo.

The same directory contains a README.txt, which points to the file `how_to_install_ecflow_on_leonardo.txt`. The latter contains instruction on how to create the virtual environment and install ecFlow on Leonardo.

Using a terminal in which the environment has been activated, navigate to the directory `ecflow_suite`. Once there, the suite can be created with the following command:

- `python define_suite_scenario_4.py --fc_date=2023051600`
  in the case of `scenario_4_da`.
- `python define_suite_scenario_4.py --fc_date=2023051512`
  in the case of `scenario_4_s3`.

Note that in `scenario_4_s3`, the suite uses the default S3 bucket name, defined inside the script `scenario_4_s3/ecflow_suite/scripts/download_from_s3.py` .
To use a different one, it specify it as one of the input arguments to the download script inside the file `scenario_4_s3/ecflow_suite/ecf_files/download_from_s3.ecf` .

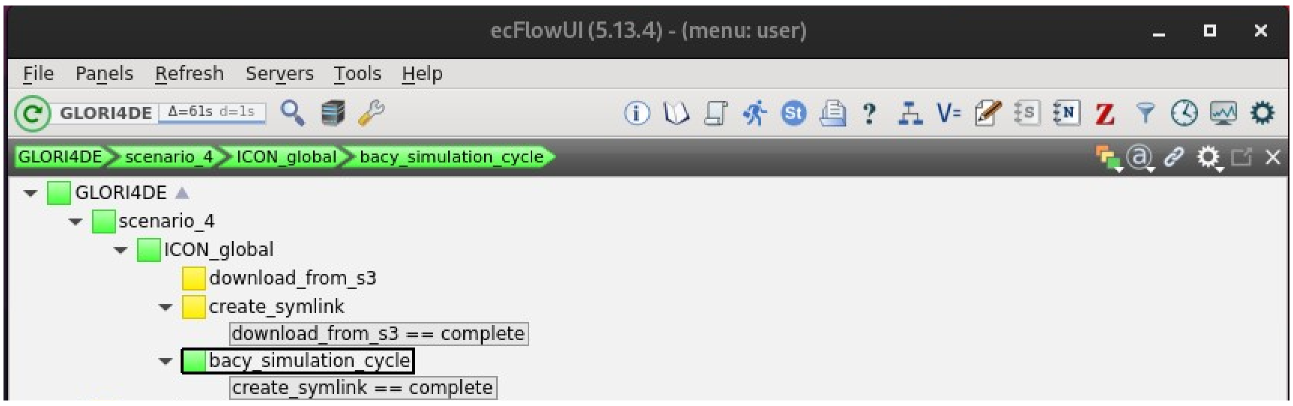
The output suite, `GLORI4DE_scenario_4.def`, is created inside the directory `ecflow_suite`.

4.3 Suite execution and monitoring

Upload the suite to the server:
`./ecflow_client --load=./GLORI4DE_scenario_4.def`
This command uses the symbolic link previously created inside the directory `ecflow_suite`.

The suite will appear in the UI, with all tasks grayed out.
To start it, execute:
`./ecflow_client --begin=./scenario_4`

Once During execution, the suite will appear as shown in Figure 1.



```
GLORI4DE
  scenario_4
    ICON_global
      download_from_s3
      create_symlink
        download_from_s3 == complete
      bacy_simulation_cycle
        create_symlink == complete
```

Figure 1: scenario_4_s3 execution as seen in ecflow_ui.

It is possible to monitor the execution by right clicking on a task and clicking on “Output” in the drop-down menu. The same window allows the visualization of the job script, as well as the modifications of the variables of each task.

The output text files of the tasks are also accessible at the following path:
`scenario_4_[s3` or `da]/ecflow_suite/ecf_files/scenario_4/ICON_global`

Once the execution of the suite is complete, the output of the ICON model is available at:
`scenario_4_[s3` or `da]/global/glori_leonardo/data`

The configuration provided in this tutorial produces a global run at 13 km resolution, with a nest over Europe at 6.5 km resolution. The extents of the two domains are illustrated in Figures 2 and 3, which show the 3-hours precipitation taken from an example run.

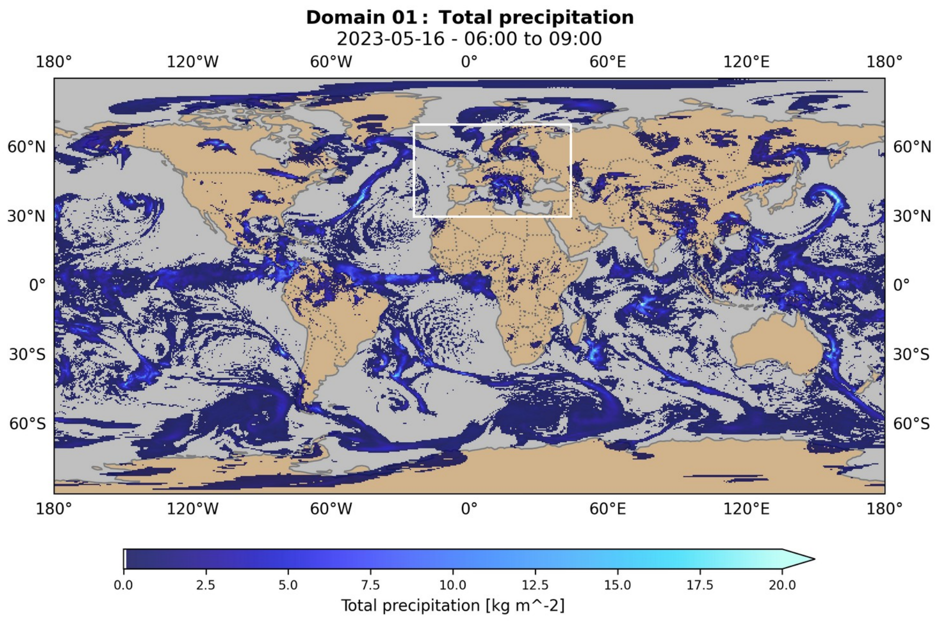

Domain 01 : Total precipitation
2023-05-16 - 06:00 to 09:00
180°
120°W
60°W
0°
60°E
120°E
180°
60°N
30°N
0°
30°S
60°S
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
Total precipitation [kg m^-2]

Figure 2: Total precipitation between 06:00 and 09:00 UTC on 16/05/2023 over the global domain.

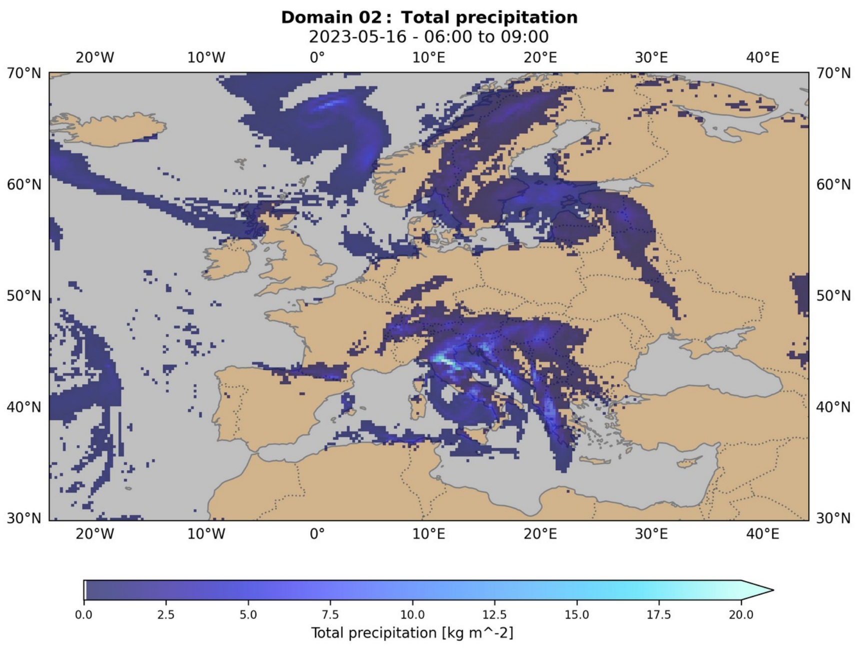

Domain 02 : Total precipitation
2023-05-16 - 06:00 to 09:00
20°W
10°W
0°
10°E
20°E
30°E
40°E
70°N
60°N
50°N
40°N
30°N
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
Total precipitation [kg m^-2]

Figure 3: As figure 2, but for the European domain.

5. Scenario 2

Scenario 2 also has 2 implementations:

- S3-based (`scenario_2_s3`)
    - Uses data transferred from DWD's Horeka HPC via Amazon S3,
    - Does not perform data assimilation,
    - Executes LAM runs by calling individual BACY modules directly from ecFlow.

- With data assimilation (`scenario_2_da`)
    - Uses IC/BC and observations stored on Leonardo, provided by DWD
    - Performs data assimilation via BACY cycle

Both implementations follow the same workflow:

1. Create the ecFlow suite,
2. Upload and start it on `ecflow_server`,
3. Monitor the execution via `ecflow_ui`.

5.1 Scenario structure

The parent directory (`scenario_2_s3` or `scenario_2_da`) contains:

- `bacy_data`, which stores inputs for BACY (IC/BC, grids, observations).
  Sub-directories:
    - `data` : Initial and boundary conditions (auto-populated by ecFlow in the `_s3` version)
    - `routfoxfox` : Grid files (exact path set in `BA_ICON_GRIDDIR` in `bacy_conf.sh`)
    - `obs_local` : Observations (used only in `_da` for assimilation)

- `ecflow_suite`, which generates and manages the ecFlow suite:
    - `ecf_files` : Task definition scripts (`.ecf` files, header, tail)
    - `scripts` : S3 download utilities (`_s3` only)
    - `define_suite_scenario_2.py` : Suite generation script

- `lam`, which contains the BACY instances. The relevant modules, whose configuration has been included in this repository, are:
    - `modules/common` : Common configuration among modules
    - `modules/core` : Data assimilation (`_da` only)
    - `modules/int2lm` : Remapping of IC/BC
    - `modules/more` : LAM execution
    - `modules/cycle` : Workflow orchestration (`_da` only)

5.2 Creating the ecFlow suite

Activate the Python environment that has the ecFlow package:

```
source tools/venv_ecflow/bin/activate
```

Then, the suite can be created by navigating to the `ecflow_suite` directory of the chosen scenario and executing the following command:

- `python define_suite_scenario_2.py --fc_date=2024101814`
  for `scenario_2_da` ,
- `python define_suite_scenario_2.py --fc_date=2023051600`
  for `scenario_2_s3` .

The output suite definition files are:

- `GLORI4DE_scenario_2_da.def` for `scenario_2_da` ,
- `GLORI4DE_scenario_2_s3.def` for `scenario_2_s3` .

They are stored inside the `ecflow_suite` directory of their respective scenario.

Note the following key differences from scenario 4:

- in `scenario_2_s3` the suite calls individual BACY modules instead of cycle,
- in `scenario_2_da` the observation directory is `obs_local` instead of `obs_global`.

5.3 Executing and Monitoring the Suite

The suite can be uploaded to the server via:

- `./ecflow_client --load=GLORI4DE_scenario_2_da.def` (for `scenario_2_da`),
- `./ecflow_client --load=GLORI4DE_scenario_2_s3.def` (for `scenario_2_s3`).

This command assumes that a link to `ecflow_client` exists inside the `ecflow_suite` directory.

Once uploaded, the suite can be started by executing:

- `./ecflow_client --begin=scenario_2_da` (for `scenario_2_da`),
- `./ecflow_client --begin=scenario_2_s3` (for `scenario_2_s3`).

The execution of the suite can be monitored via `ecflow_ui`, as described for `scenario_4`.

After execution, the output of the ICON model is available at:
`scenario_2_[s3` or `da]/lam/glori_leonardo/data`

The configuration provided in this tutorial runs on a parent domain over the Alps at approximately 2 km resolution, with a nest at 1 km resolution. The extents of the two domains are shown in Figures 4 and 5, which display the 1-hour precipitation from an example run.

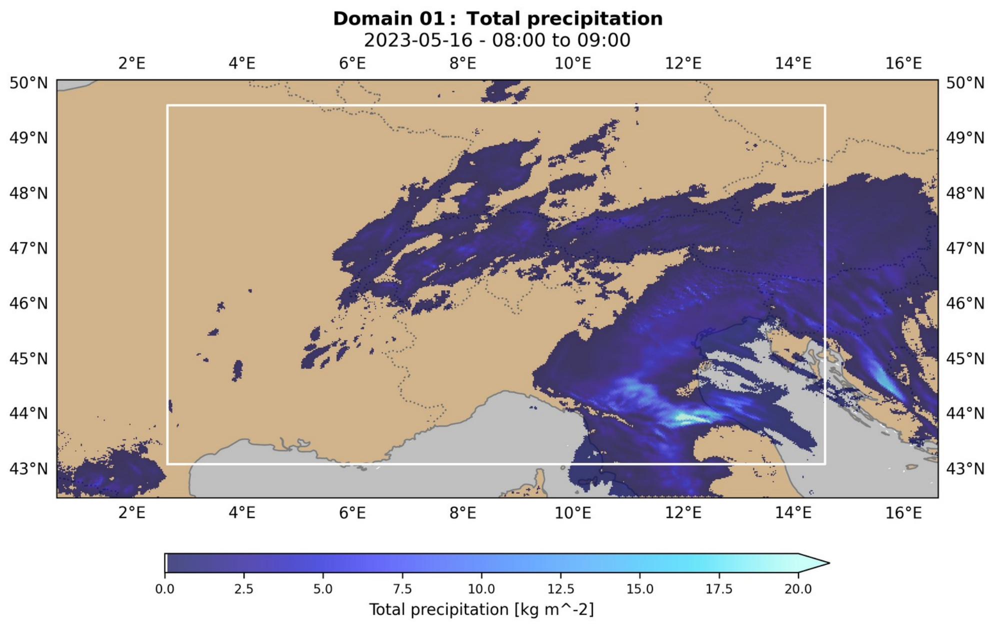


Figure 4: Total precipitation between 08:00 and 09:00 UTC on 16/05/2023 over the parent domain of the LAM.

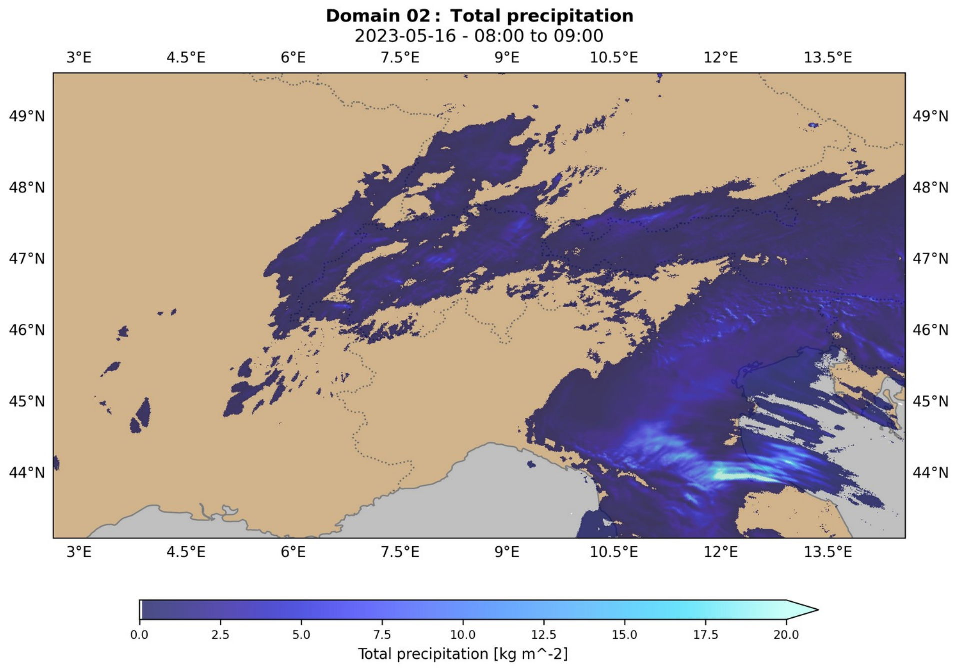


Figure 5: As Figure 4, but for the nested domain.

6. Scenario 1

Scenario 1 combines elements from Scenarios 4 (global) and 2 (LAM):

1. S3 Download:
    - Downloads the IC and analysis increments produced by DWD from Amazon S3

2. Global Run:
    - Analogous to `scenario_4_s3` (no data assimilation)
    - Uses BACY `cycle` for ICON global execution

3. LAM Execution:
    - Calls individual BACY modules (like `scenario_2_s3`)
    - No S3 download (remaps the IC/BC from the European nest of the global run)

A single ecFlow suite is associated with this scenario. However, in contrast to the previous suites, the current one contains two families. The first is dedicated to the global run, the second one to the LAM. Each family is similar in structure to their respective counterparts described for the previous scenarios.

Therefore, given the similarity between the components of this scenario and the previous two, we will refer to these sections of the tutorial for additional details on the instruction provided here.

3.1 Scenario structure

The parent directory (`scenario_1`) contains:

- `bacy_data`
  Whose content in this repository is:
    - `data` : initial conditions and analysis increments retrieved via S3 are stored,
    - `routfoxfox` : grids (path specified by `BA_ICON_GRIDDIR` in `bacy_conf.sh`).

- `ecflow_suite`
  Whose content is:
    - `ecf_files` : tasks for the global and LAM families,
    - `scripts` : Python script for the S3 download,
    - `define_suite_scenario_1.py` : ecFlow suite generation.

- `global`
  Which contains the BACY instance for the global run (mirrors `scenario_4_s3`)

- `lam`
  Which contains the BACY instance for the LAM run (mirrors `scenario_2_s3`)

3.2 Workflow Steps

First, activate the Python environment:

```
source tools/venv_ecflow/bin/activate
```

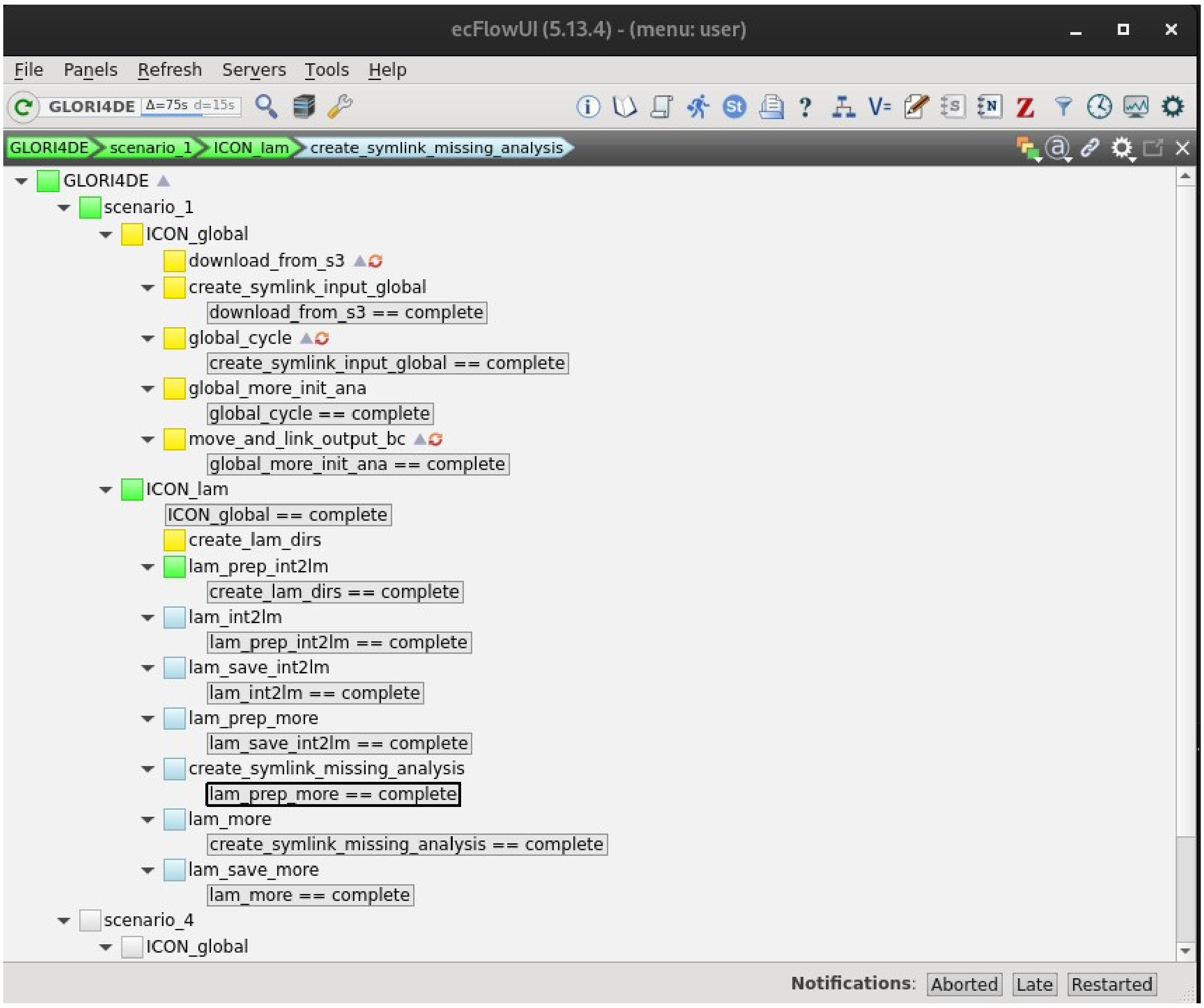
Then, navigate to the directory `ecflow_suite` and generate the suite via:
`python define_suite_scenario_1.py --fc_date=2023051512`
Note that the date is the same as `scenario_3_s3`, since the same S3 bucket is used.

The output suite, saved inside `ecflow_suite`, is named `GLORI4DE_scenario_1.def`.

Finally, the suite can be uploaded to the server and started with the following commands:

```
./ecflow_client --load=GLORI4DE_scenario_1.def
./ecflow_client --begin=scenario_1
```

Figure 6 illustrates how the suite appears in `ecflow_ui` during execution. Note the division into two families, and the different way in which BACY is used between the global and LAM run.



ecFlowUI (5.13.4) - (menu: user)
File
Panels
Refresh
Servers
Tools
Help
GLORI4DE
Δ=75s d=15s
GLORI4DE > scenario_1 > ICON_lam > create_symlink_missing_analysis
Notifications:
Aborted
Late
Restarted

```
GLORI4DE
  scenario_1
    ICON_global
      download_from_s3
      create_symlink_input_global
        download_from_s3 == complete
      global_cycle
        create_symlink_input_global == complete
      global_more_init_ana
        global_cycle == complete
      move_and_link_output_bc
        global_more_init_ana == complete
    ICON_lam
      ICON_global == complete
      create_lam_dirs
      lam_prep_int2lm
        create_lam_dirs == complete
      lam_int2lm
        lam_prep_int2lm == complete
      lam_save_int2lm
        lam_int2lm == complete
      lam_prep_more
        lam_save_int2lm == complete
      create_symlink_missing_analysis
        lam_prep_more == complete
      lam_more
        create_symlink_missing_analysis == complete
      lam_save_more
        lam_more == complete
  scenario_4
    ICON_global
```

Figure 6: scenario_1 suite execution as seen in `ecflow_ui`.

The outputs are produced into the following directories:

- the outputs of the global run (parent domain at 13 km resolution, nest at 6.5 km) are placed in `global/bacy/data` once the first family in the suite completes its execution,
- the outputs of the LAM run (parent domain at 2.2 km resolution, nest at 1.1 km) are placed in `lam/bacy/data` once the suite completes its execution.

7. Scenario 3

Scenario 3 differs from the previous ones for two main reasons:

- IFS is used for the initialization of the model
- nwprun is used alongside BACY to perform the tasks of the scenario

The output of this scenario has been chosen for the brief validation presented in the report that accompanies this repository as the GLORI4DE deliverable. Therefore, given its structural differences and the role of its results, this section provides additional details on setting up the event on 19 October 2024, which comprises:

- Continuous data assimilation from 16 Oct 2024 to 19 Oct 2024 at hourly intervals, using a 20-member ensemble,
- Ensemble forecast runs (20 members) every 24 hours at 00:00 UTC between 17 Oct 2024 and 19 Oct 2024, with a 48 h duration.

These two tasks are driven by two different ecFlow suites:

- The DA suite (data assimilation) is generated and managed by nwprun
- The forecast suite (BC remapping + LAM run) is generated by a Python script, as in previous scenarios

7.1 Scenario structure

The parent directory (`scenario_3`) contains three top-level components:

- `arkimet`
  Contains observations and IFS data. Sub-directories:
  - `archive` : storage of the data assimilation output,
  - `gts_bufr_conv` : observations in bufr format,
  - `hres_am_foricon` : deterministic IFS data,
  - `ifsens_am_foricon` : ensemble IFS data,
  - `import_signal` : notification of available data for arkimet data loader.

- `bacy`
  Similar to the parent folder of previous scenarios. It contains:
  - `bacy_data`, for storing the data needed to initialize the model,
  - `ecflow_suite`, used to generate the second suite (BC remapping, model run).
    Sub-directories:
    - `ecf_files` : ecf files used by the tasks of the suite,
    - `scripts` : auxiliary script for data movement and date/hour formatting,
    - `define_suite_scenario_3.py` : suite generation script;
  - `lam`
    Contains 2 BACY instances:
    - `bacy_remap` : BACY set up for BC remapping,
    - `bacy_more` : BACY set up for the LAM run.

- `nwprun`
  Configuration and suite generation of nwprun. It contains:
  - `conf` : HPC-specific configuration and namelists,
  - `ecflow` : suite generation and job scripts for task submission.

7.2 Data preparation

Before initializing the scenario, it is necessary to populate several empty directories with IFS files and observations, as explained by the following instruction.

1. General overview
   Data must be placed inside the following sub-directories:
   - `scenario_3/arkimet/hres_am_foricon` : deterministic IFS,
   - `scenario_3/arkimet/ifens_am_foricon` : ensemble IFS,
   - `scenario_3/arkimet/gts_bufr_conv` : observations in BUFR format.

2. Directory structure
   Within each of the two IFS directories, create year sub-directories (`2024`) containing daily GRIB files. For example:
   `scenario_3/arkimet/hres_am_foricon/2024/10-16.grb`
   Likewise for the observations:
   `scenario_3/arkimet/gts_bufr_conv/2024/10-16.bufr`

3. Import signals
   Empty IMPORTED files should be created so the DA suite knows which data are available. The structure for each sub-directory is:
   - `arkimet/import_signal/hres_am_foricon/YYYYMMDDHH00/IMPORTED`
   - `arkimet/import_signal/ifens_am_foricon/YYYYMMDDHH00/IMPORTED`
   - `arkimet/import_signal/gts_bufr/YYYYMMDDHH00/IMPORTED`

   `YYYYMMDDHH` should be replaced with the date/hour of each file (e.g. `2024101600`).

7.2 Data assimilation

The DA suite is created and driven by nwprun:

- The configuration files are:
  - `scenario_3/nwprun/conf/conf.sh`
  - `scenario_3/nwprun/conf/prod/scenario_3/conf.sh`
  - `scenario_3/nwprun/conf/prod/scenario_3/enda_nest/conf.sh`

  Variables set in files inside a sub-directory overwrite the value of the same variable from the parent directory.

- The namelists used by the tasks are located inside the directories:
  - `scenario_3/nwprun/conf/prod/scenario_3`
  - `scenario_3/nwprun/conf/prod/scenario_3/enda_nest/`

- The job script used by KENDA and a header for the submission to the Leonardo queue (which differ from the standard nwprun setup) are located respectively at:
  - `scenario_3/nwprun/ecflow/jobs/kenda.ecf`
  - `scenario_3/nwprun/ecflow/include_slurm/leonardo/sched_accounting.h`

To generate the suite, navigate to the directory `scenario_3/nwprun/ecflow` and execute the following command:

```
./ec_wrap ./suite_scenario_3_enda.py --fc_date=20241016
```

You will be prompted to write/replace the suite on the ecFlow server, answer “`y`” to both.

Start the suite with the command:
`./ecflow_client --begin=/scenario_3_enda_20241016`

In the UI, browse to:
`scenario_3_enda_20241016/day/hour_00/check_run`
then right-click `can_run` and force the status to “complete” to start the hourly cycle. The suite will continue automatically, computing the IC and analysis increments every hour until manually stopped. Alternatively, it is possible to stop the suite at the end of each day, and manually repeat the suite generations and execution for all dates until 18/10/2024.

The suite automatically saves, at the hour 00:00 and 12:00, the files necessary for the initialization of the ensemble forecasts.
These files can be found in the directory:
`scenario_3/arkimet/archive/YYYYMMDDHH/`
The string `YYYYMMDDHH` should be replaced with the date/hour of each file (e.g. `2024101600`).
The suite also retains smaller subsets for intermediate hours.

7.4 Forecast

The second suite handles BC remapping and the LAM run using BACY, mirroring the workflow of `scenario_2_s3`:

1. Activate ecFlow environment:
   `source tools/venv_ecflow/bin/activate`

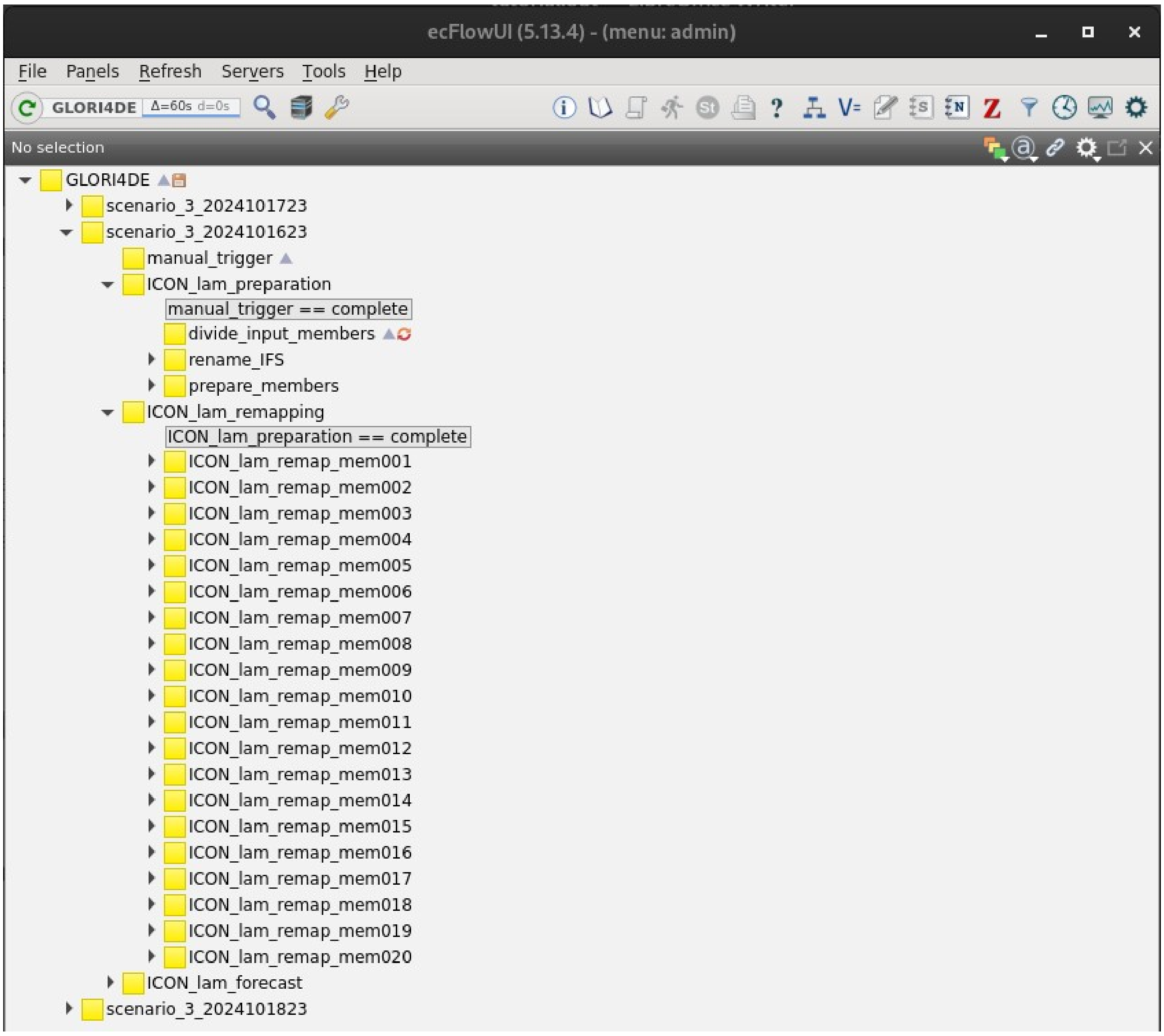
2. Generate the suite:
   Navigate to source `scenario_3/bacy/ecflow_suite` and execute:
   `python define_suite_scenario_3.py --fc_date=2024101623`
   This writes `GLORI4DE_scenario_3_2024101623.def` in the same directory.
   Replace the date with `2024101723` and `2024101823` to create the ensemble LAM run presented in the report that accompanies this repository.

   Note: The forecast start at 00:00 UTC, but the input to the Python script specifies a date/time 1 hour earlier to match the IC creation time.

3. Upload and begin the suite execution:
   `./ecflow_client --load=GLORI4DE_scenario_3_2024101623.def`
   `./ecflow_client --begin=scenario_3_2024101623`

Once both suites have completed, your BACY outputs (LAM forecasts and remapped BC) will be available under:
`scenario_3/bacy/global/glori_leonardo/data`

Figure 7 provides an example of how the suite appears in `ecflow_ui` after execution. The suite is divided into three families. Calls to the BACY modules are performed independently for each ensemble members, as exemplified by the family that performs the remapping of the BC.



```
GLORI4DE
    scenario_3_2024101723
    scenario_3_2024101623
        manual_trigger
        ICON_lam_preparation
            manual_trigger == complete
            divide_input_members
            rename_IFS
            prepare_members
        ICON_lam_remapping
            ICON_lam_preparation == complete
            ICON_lam_remap_mem001
            ICON_lam_remap_mem002
            ICON_lam_remap_mem003
            ICON_lam_remap_mem004
            ICON_lam_remap_mem005
            ICON_lam_remap_mem006
            ICON_lam_remap_mem007
            ICON_lam_remap_mem008
            ICON_lam_remap_mem009
            ICON_lam_remap_mem010
            ICON_lam_remap_mem011
            ICON_lam_remap_mem012
            ICON_lam_remap_mem013
            ICON_lam_remap_mem014
            ICON_lam_remap_mem015
            ICON_lam_remap_mem016
            ICON_lam_remap_mem017
            ICON_lam_remap_mem018
            ICON_lam_remap_mem019
            ICON_lam_remap_mem020
        ICON_lam_forecast
    scenario_3_2024101823
```

Figure 7: Second suite of `scenario_3` after the execution as seen in `ecflow_ui`.

8. Visualization

To explore and analyze the model output produced by BACY, two Jupyter notebooks have been provided in this repository. They support visualization on both the ICON native triangular grid and the remapped data on a regular lat/lon grid. These notebooks are located in the directory `visualization.`
They can be used with the ICON forecast generated by BACY in any scenario, if correctly configured (e.g. paths, grids).

Two types of visualizations are supported:

- Interactive plots (Panel-based) for variables on the native grid,
- Static images for surface-level variables on the regular grid.

To verify the notebooks are functioning correctly, we provide a set of sample output files (for both LAM and global configurations) available via Google Drive. These are not required for standard use but can be helpful to confirm that your setup works.

For detailed setup instructions, including required Python packages, how to download and where to place the example files, and how to run the notebooks both locally and on a remote server, please refer to the file: `visualization/README.txt`.