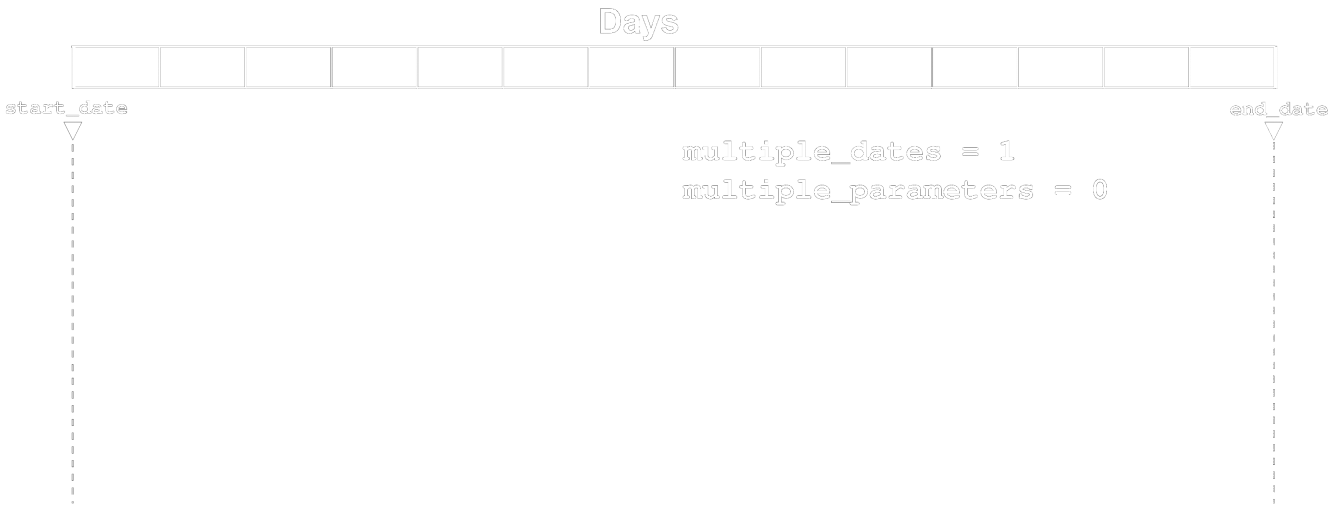
Table of Contents

The user only needs to edit the **`experiment.wrf4g`** file in order to design the experiments. From this file WRF4G generates all the WRF configuration files (namelist.wps and namelist.input) needed to run the simulations planned for the experiment, and then launchs them. The namelist.input files are constructed modifying the default namelist provided by NCAR with WRF source code. experiment.wrf4g has the following structure:



- **experiment_name:** Name of the experiment. **It has to be unique**. Duplicity of experiment names would give different WRF4G answers:

Domain of simulation

- **max_dom:** Number of domains.
- **domain_name:** Name of the folder with the information about the domain of the simulations, this is, the files generated by geogrid.exe ('`nameslit.wps`' & '`geo_em.d[nn].nc`'). This folder needs to be located at `${WRF4G_DOMAINPATH}` of the resources.wrf4g file.

Input data

- **extdata_vtable:** Vtable of the `ungrib` module to be used to decode provided GRIB input data.
- **extdata_preprocessor:** Name (just the ending `[NAME]` section in '`preprocessor.[NAME]`') of the necessary pre-processor useful to make the specific input data available for WRF model. User could not be interested on the permanent WRF-necessary modification of any source of data. With the specification and design of a pre-processor, necessary WRF modifications (g.e.: ASCII to GRIB, variables re-coding, complete input data). Pre-processors are included in `WRF4G-1.0beta.tar.gz` tar file in '`${WRF4G_LOCATION}/repository/apps`'
- **extdata_path :** Path to the input data. It must be consistent with the pre-processor design.
- **extdata_interval:** Interval of the input data (in seconds). On multiple input data sources use the smallest one.

Output data

- **postprocessor:** user might be interested in the transformation of the WRF output files. A first generic post-process of the output will be automatically carried out if a valid name `postprocessor.[NAME]` is provided. Post-processors are included in the `WRF4G-1.0beta.tar.gz` tar file at '`${WRF4G_LOCATION}/repository/apps`'

Experiment time-specification

- **start_date:** Beginning of the experiment
- **end_date:** End of the experiment.
- **chunk_size_h:** Length (in hours) of the smallest independent temporal piece of the experiment. It will be the same for all the realizations of the experiment, and at the end of each one a `wrf-restart` file will be written , in order to continue the experiment with the consecutive chunk

  multiple_dates: Binary flag (0:No, 1:Yes) indicating whether the experiment will be based on independent realizations started at different dates with the same length. When activated it has to be accompanied with appropriated values of:
  - simulation_interval_h: interval (in hours) between independent realizations. It can be specified as multiple values, as follows: '`N*chunk_size`'
  - simulation_length_h: length (in hours) of the independent realizations. It can be specified as multiple values, as follows: '`M*chunk_size`'

- **rerun:** Binary flag (0;No , 1: Yes) indicating whether the experiment should be rerun, if the experiment has already run (over-writing output files and data-base values)

  multiple_parameters: Binary flag (0;No , 1: Yes) indicating whether the experiment is based on independent realizations with different values on the WRF-namelist.input file. When activated it has to be accompanied with appropriated values of:

  - multiparams_variables: The set of parameters that we want to change (some of them or all) between realizations. It is given as a coma-separated WRF-namelist.input files (no record specification is necessary if the parameter already appears on the provided WRF-version namelist.input template?, If not it should be `[parameter]__[record]`)
  - multiparams_combinations: Values of the changing parameters for each realization. Parameters should be separated by commas and realizations are separated by '`/`'.
  - multiparams_nitems: number of values that should have the namelist parameter. `${max_dom}`, as many values as domains; `1`, a single value.
  - multiparams_labels: the output of the experiment is organized with a folder for each realization. The pattern of the folder name for a given realization is `[start_date]_[multiparams_combinations]('_' separated)`. However, user can provide a set of labels ('`/`' separated) for each realization, and then folder names will be '`[start_date]_[multiparams_labels]`'

Debugging

- **clean_after_run:** binary flag (0: no, 1: yes) indicating whether the heavy-stuff of the simulation (g.e.: `wrf.exs, rsl.`) should be removed from '`${WRF4G_RUN_LOCAL}`'. The maintenance of these files on running place could be desirable for debugging purposes. (Default value is 1)
- **save_wps:** binary flag (0: no, 1: yes) indicating whether boundary and initial conditions (`real.exe` output) should be preserved. They will be used if the experiment launched again. (Default value is 0)

Parallel configuration

- **real_parallel:**binary flag (0: no, 1: yes) indicating whether the `real.exe` binary is compiled in serial or parallel mode (Default value is 0)
- **wrf_parallel:** binary flag (0: no, 1: yes) indicating whether the `wrf.exe` binary is compiled in serial or parallel mode (Default value is 1)

WRF-namelist parameters

User provided namelist values. They will be over-written on `multiple_parameters` experiments (only the included parameters). User can modify any of the parameters of the namelist. The name must be the same of the `namelist.input` entry, with a prefix-flag and the record to where belongs.

- **Single valued; NI_[namelist_parameter_name]_ _[record]:** `NI` flag indicating that [namelist_parameter_name] of record [record] has a single value.
- **One value per all domains; NIN_[namelist_parameter_name]_ _[record]:** `NIN` flag indicating that [namelist_paramter_name] of record [record] has the same value for all the domains of the experiment.
- **One value per domain; NIM_[namelist_name]_ _[record]:** `NIM` flag indicating that [namelist_paramter_name] of record [record] has a different value for each the domains of the experiment.

Note that the namelist variables already present in the default namelist.input do not need to be provided with _ _[record].

Others

- **timestep_dxfactor:** If present, the time step is computed as dx***timestep_dxfactor**, in kilometers. Defaults to 6, as suggested by the WRF team for most applications. Under some circumstances (cfl problems) a lower value may be needed. In any case, the time step is adjusted to the higest value lower than `timestep_dxfactor` times `dx` fitting evenly in a 1 hour period.

Examples

Multipledates runs

When the `multiple_dates` parameter is set to 0, two additional configuration options (**simulation_interval_h** and **simulation_lenght_h**) are available to set up hindcast runs. Their usage is best understood through a couple of examples. Imagine you are interested in running a 7-day forecast starting every day at 12Z for the period from 1-Jan-2001 to 31-Dic-2001. This can be accomplished by setting:

```
start_date = "2001-01-01_12:00:00"
end_date = "2002-01-01_00:00:00"
is_continuous = 0
 simulation_interval_h = 24
 simulation_lenght_h = 168 # this is 7 days in hours
```

and the `wrf4g_submit` will setup 365 independent runs performing the 7-day forecasts, which will be distributed in the grid or in your local cluster.

Multipleparameters runs

Multi-physics runs are activated when the `multiple_paramters` parameter is set to one. Three additional parameters allow you to configure the physics combinations of your interest.

This mode allows varying parameters in general, not necessarily physics options. For instance, if user could be interested in checking WRF performance on domain MPI-decomposition, by settting:

```
is_multiphysics = 1
 multiphysics_variables = "nproc_x,nproc_y"
 multiphysics_nitems = "1,1"
 multiphysics_combinations = "1,4/4,1/2,2/-1,-1"
```

you can vary the domain decomposition and send an experiments with four realizations, and the only difference is the way the MPI domains are partitioned.