

## Reference system status February 2002

*Per Undén, Gerard Cats*

*15 February 2002*

## Release notes of HIRLAM 5.1.3

*Gerard Cats ([cats@knmi.nl](mailto:cats@knmi.nl))*

HIRLAM 5.1.3 implements:

1. the [ISBA](#) surface parametrisation and analysis scheme.
2. extensions to the [verification package](#)
3. [technical improvements](#)

The changes concern [source libraries](#). In particular the ISBA changes are so extensive that they could not be implemented as an 'option'. So from version 5.1.3 onwards, it is not possible not to use the ISBA scheme.

### **Meteorological impact**

Only the new surface package has meteorological impact.

The package, as referred to here, consists in fact of two developments, namely a tiling surface scheme using ISBA for the parametrisation of the land surface processes (Avisar and Pielke, 1989; Noilhan and Planton, 1989) and an assimilation module for surface parameters including soil temperatures and water contents, 2m temperature and relative humidity over land, snow depth, SST and ice fraction. The soil moisture content is sequentially assimilated according to the method proposed by Mahfouf (1991). The HIRLAM implementation closely follows Arpège (Giard and Bazile, 2000). The postprocessing has been enhanced with new variables corresponding to the different tiles in addition to the grid averaged values.

The impact of the new surface package has been the subject of many investigations, *e.g.*

- "The ISBA scheme for HIRLAM-5: latest results prior to the operational implementation" by Ernesto Rodríguez, Simo Järvenoja, Beatriz Navascués and Juan José Ayuso, in [HIRLAM newsletter 38](#)
- "The tiling surface scheme for HIRLAM5: features and latest results" by Ernesto Rodríguez, Beatriz Navascués and Juan José Ayuso, in Proceedings of the HIRLAM5 Workshop on surface processes, turbulence and mountain effects (Madrid, 22-24 October 2001) (published January 2002).
- "ISBA tests in a Nordic area" by Simo Järvenoja, in Proceedings of the HIRLAM5 Workshop on surface processes, turbulence and mountain effects (Madrid, 22-24 October 2001) (published January 2002).
- "The new HIRLAM5 surface analysis" by Beatriz Navascués, Juan José Ayuso and Ernesto Rodríguez, in Proceedings of the HIRLAM5 Workshop on surface processes, turbulence and mountain effects (Madrid, 22-24 October 2001) (published 2002).

The main impact, in terms of verification scores, is a marked reduction in *bias* and *rms* error of screen level parameters, 2m temperature and relative humidity.

## Description of the changes

### the ISBA scheme

For the new parametrisation scheme, many new routines have been included in the library `phys`. Also, quite a few routines in library `grdy` have been changed. Because many new surface fields are now required, the climate file generation has been revised (although most of the changes were already in the script `Climate`, under the `'ISBA'` option). This implies that **to run 5.1.3 you cannot use old climate files; recreate them, and for the same reason, you must use new boundary files**. The postprocessing scheme, in particular for near-surface parameters had to be revised (library `prpo`).

The surface analysis suite has been completely redesigned. It now consists of one single execution of program `'span'`; the scheme has been collected in library `span`. Its actions are:

- sea-surface temperature analysis. If no other data are available this analysis is based on ECMWF's SST analysis
- Optimum interpolation of temperature and relative humidity at screen level, over land
- The Mahfouf (1991)) scheme to assimilate soil water content over each land tile
- The Giard and Bazile (2000) method to correct soil temperatures over each land tile
- Successive correction method for tile average snow depth
- Diagnostics of water and ice fractions based on SST analysis.

### extensions to the verification package

Ben Wichers Schreur wrote about these:

#### Modifications 20010904

##### Author

Ben Wichers Schreur, KNMI ( [ben@knmi.nl](mailto:ben@knmi.nl) )

##### Synopsis

This set of modifications supports extensions to the verification system:

- introduction of 6, 12 and 24 hour rain rate verification, both as areal averaged statistics and as contingency tables.
- Introduction of a masking tool, to allow an arbitrarily complex selection of data for verification, e.g. only model points over land, only model points above 1000 m, only when temperatures are in the range  $-4: +4^{\circ}\text{C}$ .
- introduction of field verification (which is switched on by the string `:fd:` in `VERIFY` in `scripts/Env_expdesc`)

## single field verification results

### Field verification (files fsc)

The files `fsc` contain field verification results. Fields are verified against the verifying analysis. With each field, a reference field is defined. Currently we use persistence for this, and to be precise: the initialised analysis at the beginning of the forecast.

### Precipitation contingency tables (files pct)

The first line describes the parameter; the listed date/time is the observation time. Then we see the applied class limits, followed by the number of observed and grid points per class. Grid points are only those that are nearest to an observation. Some moments for the observation and grid point distributions are also printed. The contingency table has the observation classes along the x-axis. So in this case, the forecast (+48) has many points with light to moderate precipitation near observations reporting less than 0.1 mm in 6 hours.

### Accumulated statistics

Field verification statistics are automatically accumulated per month. These accumulated statistics are valuable to quickly get a quality measure of an experiment covering many cycles. They are obtained by adding the latest results to previous accumulations. They have little relevance for experiments consisting of one or two cycles (be aware that they may even be wrong, namely if you forgot to remove the accumulations of earlier runs with the same cycle: that cycle will be counted several times).

The files `cfl1` contain the model climate (i.e. the average model state over the accumulation period). The files `efl1` contain two fields per model field, the first being the *bias*, the second the *rms*, of the comparison against the analyses. (In here, `l1` is the forecast length). The files are kept in the archive, with title `FVyyyyymm_hh.tar` (so the usual `CYCLEDIR` format, but with the day indicator dropped).

---

## Technical improvements

Mini-SMS has been changed. It used to stop submitting jobs when any task had failed. But now it will continue to submit tasks. Of course, only tasks that do not depend on the completion of the failed tasks will be submitted. When after an abort mini-SMS detects that no further tasks can be submitted, and no tasks are running anymore, it will abort, unless the graphical monitor is on.

Several new 'Actions' have been added. These have been implemented as Perl subroutines, in script `Actions.pl`. The following actions are new:

`cleanup`

Remove files from all locations where they may exist. Because mini-SMS supports (if not to say favours) the use of a network of computers, files may be left on many disks. The following will remove all files (and also all directories):

```
Hirlam cleanup -ALL -go
```

The following will remove all observation, analysis, and `mf` (lateral boundary) files:

```
Hirlam cleanup -go REMOVE:ob,mf,an
```

Without the `-go`, files will not be removed, but listed. This is useful if you want to first

verify that not too many files will be thrown away.

`cleanup` will not affect your main experiment working directory `HL_WD` (unless you totally mixed up this directory with other directories, but I guess in that case you'd better throw away everything anyway).

**Please use `Hirlam cleanup -ALL -go` when you think that your experiment has lost its value**, so as to keep disks and archives as clean as possible. Note that you always can rerun the experiment, because `HL_WD` will be kept. Of course, you should also remove `HL_WD` once you have lost interest in the experiment.

`echo` or `print`

print the values of variables; examples:

```
Hirlam echo
```

Print all environment variables

```
Hirlam echo -perl HL_DATA HL_EXP
```

Print `HL_DATA` and `HL_EXP` in Perl format (also available: `-sh` and `-csh`).

`locatesource`

Try to find where the mentioned source file(s) may reside.  
In which libraries do I find `COMDDR` and `POSTP`? Type:

```
Hirlam locatesource comddr postp
```

`prod`

This action has been designed with operational installations under mini-SMS in mind. It is intended to be called fairly frequently, probably best from `crontab`. It will normally invoke ``Hirlam continue'` (but ``Hirlam start'` if there is no `HL_WD/progress.log` yet). However, if `prod` discovers that there is already a mini-SMS running, it will do nothing.

This makes it safe against multiple submissions, and thus allows it to be called frequently (although better not more frequently than once per minute, because it is safe against multiple executions of HIRLAM but not completely against concurrent executions of `Prod` itself).

In operational implementations, there will probably be a complicated schedule to decide on submission, *e.g.*, to wait for the observations to arrive, but not to wait more than 3 hours; the forecast length may depend on the time of day *etc.* The action supports the inclusion of your own schedule. Create a Perl script with the schedule, store that script in `HL_WD`, and invoke

```
Hirlam prod schedule
```

where `schedule` is the name of your script (default: `schedule`). As an example, I add a possible schedule at the end of this note, as [an appendix](#). The example is an extension to the old action ``prod'`, which based submission on wall clock time. It will be a fairly small task to include *e.g.* tests on the availability of input files.

The **graphical monitor**, `mXCdp`, has been extended as well. The main change, to the user, is the implementation of the concept of **'selected node'** (similar to the 'current node' concept of full SMS). The user can select a node (one at a time) by clicking with the middle mouse button, by double-clicking with the left mouse button, or by clicking with the left mouse button to the left of the node in the main window. A selected node can be 'de-selected' by the same actions. If no node has been selected, then mini-SMS will automatically select a node that becomes 'aborted' (if there is one). (This implies that if more nodes failed, only the first one that mini-SMS diagnosed to have failed, will become the selected one).

When a node becomes selected, its display colours will be inverted. Also, its name will be displayed at the bottom of the main window, and a number of buttons will appear there. If the selected node is a task, the left-most button is labelled 'output . . .'. Pressing this will open a window with the `stdout/stderr` file of the selected task (if available). In combination with the property that an aborted task will be selected automatically, this gives a quick access to key information as to why the task failed (but be aware that possibly other tasks failed as well).

## Acknowledgements

Beatriz Navascués and Ernesto Rodríguez contributed to the description of the surface packages; Ben Wichers Schreur to that of the verification package. Xiaohua Yang revived the postprocessing of near-surface data immediately after initialisation, by invocation of a single physics time step.

---

# Release notes of HIRLAM 5.1.2

*Gerard Cats ([cats@knmi.nl](mailto:cats@knmi.nl))*

---

**Note: if you are going to install 5.1.2 locally, and you have been using an earlier version already, you must modify all older local versions of `Env_system`, see [Env\\_system](#), below.**

---

HIRLAM 5.1.2 implements some technical improvements:

4. Allow the file with lateral boundary data to be used as first-guess [on the first cycle only](#)
5. Introduce the option `PROD`, to silently exit mini-SMS when complete
6. Introduce `smsmeter`
7. Introduce `HL_LIB`, to allow scripts, executables, *etc.*, in a different directory than `HL_DATA`.
8. Simplify [HL\\_DATA processing in Env\\_system, and use R\\_HOST consistently](#)
9. Extend `PBMESR`
10. Extend [the graphical user interface mXCdp](#)
11. Allow [parallel compilations](#)

The changes concern [source libraries](#), [resource files](#), and [local installation](#) procedures.

## Meteorological impact

None.

# Description of the changes

## First-guess from lateral boundary data file

Before 5.1.2, the script `Fg` would take the first file that is mentioned in the 'boundary strategy file' to be used as first-guess in case it could not find a forecast of the appropriate length from the previous cycle. It would issue a warning message, and it would report its 'decision' in the logfile and in the cycle history file, but both files may escape inspection by the user for a long time. In the past, this has led to corrupted experiments, *e.g.* when in an experiment with 3-hours cycles the forecast was not instructed to produce a history file after 3 hours: there never was a 3 hours forecast from the previous cycle, so each cycle started from a boundary file.

From now on, the script `Fg` will abort when it cannot find the proper first-guess, except on the very first cycle of an experiment.

The first cycle of an experiment is identified by an entry in the `progress.log` file. The variable `DTGBEG` is not suitable for this, because this merely indicates from which date the current run should start, but not whether this run is a continuation of an earlier experiment. The entry in `progress.log` to identify the first cycle is written if the experiment is started with `Hirlam start`.

In this context, **you are reminded to use:**

- `Hirlam start` when you want to start an experiment, from the first cycle
- `Hirlam continue` when you want to continue an experiment

If, in the latter case, you use `start` instead of `continue`, you will forsake the test for the proper first-guess; often this will be harmless but it is a potential pitfall... (If you use `Hirlam continue`, you should specify exactly the same command line arguments as you did with `Hirlam start`, except that you should not specify `DTG`; the value of `DTG` will be derived from `progress.log`.)

The method to determine the first-guess was confusing. The file `sds` (start data set) was first set to the boundary file, and then later overwritten if there was a more appropriate first-guess. This confusion has now been removed; the first usage of the file is to define the domain to the observation pre-processing, and for this purpose now the `climate` file is used.

## Option PROD

If mini-SMS is used with the graphical interface (`mxCdp`), it used to pop up a window to warn the user that it became complete or aborted. The user had to acknowledge this, and thus take mini-SMS out of execution. If the user failed to do this, mini-SMS would die after some time (default: 24 hours). This is to allow the user to continue to use the monitor *e.g.* to view the logfiles of the run.

In operational production runs, however, this procedure is less desirable, because it required human intervention for every cycle (assuming mini-SMS is started afresh for every cycle, which procedure is now emerging as the most likely in operational systems). Therefore, mini-

SMS was extended with an option to 'auto-exit' after some time. Invoke this option by adding `PROD=1` on the command line to start or continue `Hirlam`.

It is anticipated that the option will be used later to identify an operational scenario, but currently `PROD=1` has no other effect than to make mini-SMS exit gracefully after 20 seconds after its completion.

**This is the recommended way to execute an operational HIRLAM run:**

```
Hirlam start PROD=1
```

the very first time to start operational production

```
Hirlam continue PROD=1
```

any later operational run

**smsmeter**

mini-SMS has been extended with `smsmeter` (see [the documentation](#) of SMS and of mini-SMS).

Currently, this feature is not of much use for HIRLAM. The obvious application would be to monitor the progress of the forecast, and to start postprocessing of its output files when they are ready. This would require the invocation of `smsmeter` from the forecast, which must be through a system call. However, on `vpp` at ECMWF, there is a fairly low maximum number of system calls that may be issued from a multi-processor job. This was worked-around by the so-called *Listener* mechanism, enabling postprocessing of files during the forecast run while avoiding the need to issue system calls.

Although the `smsmeter` cannot be used from the forecast itself, it can be used from the *Listener*. Scripts `Postproc` was modified to increase the meter to the forecast length of the processed file, if its unit number is 31. This unit number identifies a model history file, and its completion indicates that the forecast and the postprocessing have advanced to the indicated forecast length. In the default HIRLAM configuration, there are no actions dependent on the value of this meter, but if the suite definition file is embedded into one in which HIRLAM is just one component in an (operational) schedule, the value of the meter may be used to start other processes.

To give some functionality to this in the default configuration, the graphical monitor was extended with a slider, indicating the progress of the postprocessing. After the forecast has progressed enough to have produced the guess field for the next cycle, the background of the slider becomes green. The user can modify the value of the meter - but beware that this alters the meter value, not the progress of the forecast...

To use this new feature of mini-SMS, scripts that want to increase the meter value must have an executable `smsmeter` in their path. The HIRLAM system provides a script, in its scripts directory. To avoid confusion in cases where you want to use full SMS, the location (pathname) of `smsmeter` is provided in the environment variable `SMSMETER`, of which the value is set in `Env_system`. To invoke `smsmeter`, use the following line in the invoking script:

```
$SMSMETER metername new_metervalue
```

(so do not call `smsmeter` but call `$SMSMETER`).

## HL\_LIB

From now on the working directory to create and compile sources is `HL_LIB`. The value of this variable is set in `Env_system`, default equal to `HL_DATA`. As a consequence, scripts, object libraries, executables, *etc.*, are created and kept in `HL_LIB`. The separation of `HL_LIB` from `HL_DATA` allows to keep those files, that are semi-permanent for a particular experiment, on a different file system than the possibly volatile `HL_DATA`.

It also allows to put these files in a directory that is not private for the experiment, thus allowing re-use of them by different experiments. This will speed up the initial steps in an experiment, by avoiding compilations; but, (needless to say but yet better said) this is a very powerful source of confusion between different experiments. Perhaps better not use this...

In `Env_system`, `HL_DATA/scripts` in the search path was changed to `HL_LIB/scripts`.

## Simplified HL\_DATA and consistent R\_HOST processing

Before 5.1.2, in `Env_system` environment variables `HL_DATA_HOSTn` had to be set for each host. For a consistent introduction of `HL_LIB` similar variables `HL_LIB_HOSTn` would have been needed. To avoid this complication, the system now determines the value of these variables by executing `Env_system` with `SMSNODE` temporarily set to the host in question. For this, `scripts/hosts.h` was modified. The same mechanism is used now for `HL_DATA`, thus obviating the need for `HL_DATA_HOSTn`. Use is made of the fact that these variables are needed only by the tasks that copy files from the current host to some remote host, or *vice versa*. The variable `R_HOST` is now used consistently to identify the remote host. The choice of data to be copied (*viz.* `HL_DATA` or `HL_LIB`) is indicated by the variable `SRC_DIR`.

## PBMESR

The utility to read purely binary messages, `PBMESR`, now is able to read past messages that are too long for its internal buffers, and it now can handle up to 5 input streams.

## mXCdp

The graphical monitor now, by default, closes nodes that are queued or complete, and opens them when becoming submitted. This behaviour can be influenced with an option in the `View` menu.

It also was improved, cosmetically; this improvement made it possible to add sliders to the nodes that have meters.

## Parallel compilations

Script `Make_ref` was modified to pass the flag `$(MAKE_COMPILE)` to the `make` that creates the object libraries. This can now be used to invoke several (parallel) processes for compilations: in the environment variable `MAKE_COMPILE`, specify the number of processes per library through the `-j` option. For example, in `Env_system` add the line:

```
MAKE_COMPILE="-j 8" export MAKE_COMPILE
```

(there may be resource limits on your system that inhibit high values, *e.g.* by limiting the maximum number of temporary files simultaneously open by `make`. At ECMWF, anything  $>1$  may fail).

## Acknowledgements

Kalle Eerola suggested to use the climate file to define the area, and to implement `smsmseter` and `HL_LIB`. Toon Moene modified `PBMESR`. Ole Vignes did the cosmetic changes to `mXCdp`. Jan Boerhout suggested to compile in parallel.

## Planned releases

- 5.1.4      Conservative correction of (revised) CBR, NLDYNVD  
            Testing for a Reference release
- 5.2
- 5.2.1      3D-VAR  
            DMR with mini-SMS  
            Resolution increase of the Reference system and DMR  
            Testing for a Reference release
- 5.3