

HEFS-1.0.2 Release Notes

Release Date: 12/3/2013

Release Type: Scheduled

HEFS Build: 1.0.2

Build and Package Date: 11/1/2013

Tested against FEWS Binary: 2013.01 build 43108, patched from 42428

Introduction

This document contains release notes for HEFS-1.0.2.

IMPORTANT: This release of HEFS-1.0.2 assumes that you've already upgraded your CHPS to version 4.0.1. As part of that update, your version of GraphGen needs to be migrated to the new file system based approach for reading products and settings files. **GraphGen will not function if you do not do this.** See [FBz 1117](#) for more details. The document containing the steps for migrating GraphGen to the new products and settings has been released as part of CHPS-4.0.1. If you have not upgraded GraphGen as part of the CHPS-4.0.1 release, you will need to do this. A version of the *Graphics Generator Install Guide* can be found attached to [FBz 1078](#).

If an RFC has not already updated to HEFS-1.0.1, parameters may need to be re-estimated. The EnsPostPE and MEFPPE *parameter estimation standalone(s)* used for the previous release can be used for this release. This will save time importing data. Follow the install notes to update the binary files used for those standalones to the new release. If configuration changes since the last release are significant (e.g., new forecast points are added), necessitating that the *parameter estimation standalone(s)* be reconstructed, then see Section 5.1.1 of the EnsPost and MEFPPE configuration guides for instructions on porting over the run areas from the old standalone to the new standalone. When parameter estimation is completed for this release, save your standalone(s) in case parameters need to be re-estimated.

Below are two tables of the fixes and enhancements in this release. Following that is a list of changed documentation. Afterwards, the fixes and then enhancements are described in greater detail.

The HEFS Test RFCs are expected to test existing, fixed, and new HEFS functionality. The Test Manual provided in this release specifies what testing is expected from each RFC and contains the test procedures for many tests.

Fixes

FogBugz ID	Reported By	Title
N/A	N/A	Misleading display of reference forecasts for aggregation units
N/A	N/A	Bug in the getWidth() method of AdaptableComboBox.java
N/A	N/A	Bug in the setCellEditors() method of EditorTable.java
N/A	N/A	Bug in modality of two GUI classes
N/A	N/A	NullPointerException in SearchableComboBox.java
N/A	N/A	Spelling error in the html explanation of the Brier Skill Score
N/A	N/A	Real-valued verification thresholds not aggregated properly

N/A	N/A	Incorrect re-scaling of aggregation weights when missing first metric
N/A	N/A	Changed default behavior for “reset” buttons

Enhancements

FogBugz ID	Requested By	Title
N/A	N/A	Renamed MoreOutputOptions.java
N/A	N/A	Added functionality to allow for joint pairing
N/A	N/A	Improved organization of MoreParOptionsDialog.java
N/A	N/A	Allow real-valued bounds on computing metrics
N/A	N/A	Detection limit on forecasts and observations
N/A	N/A	Option to exit MoreInputDialog from the Forecast Scale table
N/A	N/A	Separate pairing options from the “Other options” dialog
N/A	N/A	Improved control on aggregating forecasts and observations

Documentation

The following pieces of documentation have been modified since the last release and can be found in the 'documentation' directory at the root of the package. All the HEFS documentation may be found online at <http://www.nws.noaa.gov/oh/hrl/general/indexdoc.htm>.

- *Modified:*
 - HEFS install notes
 - HEFS test manual
 - EVS user manual

Notes

Fix: Misleading display of reference forecasts for aggregation units

Description

When aggregating verification results from several locations, multiple Verification Units (VUs) are used. Each VU has a separate reference forecast for computing skill and the overall results are provided in an aggregation unit. In the EVS verification plots, the reference forecast is included in the plot title for skill metrics. For an Aggregation Unit (AU), the reference forecast is shown for the first VU that forms the aggregation. However, this was not qualified in the plot titles as being the reference forecast associated with the “first” VU only.

Cause

For AUs, a failure to identify the reference forecast in the plot titles for skill scores as being the reference forecast associated with the “first” VU that forms the aggregation.

Fix

Qualified the EVS plot titles for aggregated verification results. The reference forecast is now qualified as being the “first” of several potential reference forecasts (one for each VU forming the aggregation).

Notes

Tested visually by confirming the updated plot titles.

Fix: Bug in the `getWidestItemWidth()` method of `AdaptableComboBox.java`

Description

The `getWidestItemWidth()` method of `AdaptableComboBox.java` sets the width of the `AdaptableComboBox` dynamically depending on the width of the widest element in the box. However, the box allowed for iteration over null elements, which resulted in a `NullPointerException` when the box comprised no elements.

Cause

Failure to screen out null elements when setting the width of the `AdaptableComboBox` dynamically.

Fix

Added a check for null elements in the `getWidestItemWidth()` method of `AdaptableComboBox.java`

Notes

Tested by constructing an `AdaptableComboBox.java` with null elements.

Fix: Bug in the `setCellEditors()` method of `EditorTable.java`

Description

The `setCellEditors()` method of the `EditorTable.java` accepted generic objects as cell editors, while the internal code of `EditorTable.java` casts to `TableCellEditor.java` objects in various places, potentially resulting in a `ClassCastException`.

Cause

Failure of `setCellEditors` to take `TableCellEditor.java` objects.

Fix

Updated the `setCellEditors` method of `EditorTable.java` to take objects of class `TableCellEditors.java` only.

Fix: Bug in modality of two GUI classes

Description

The `MoreInputDialog.java` and the `MoreVerificationWindowDialog.java` did not previously allow for modality on construction. However, parameters in these dialogs are affected by choices in other dialogs. Thus, modality is required to ensure that the dialogs are always in a consistent state, i.e. that entries cannot be changed elsewhere before closing these dialogs.

Cause

Inability to set the modality of MoreVerificationWindowDialog.java and MoreInputDialog.java and to ensure the default setting for each dialog is modal (to prevent an inconsistent state arising from changes made elsewhere).

Fix

Enhanced the constructors of MoreVerificationWindowDialog.java and MoreInputDialog.java to allow for modality. Changed the default setting to modal.

Notes

Tested by checking that both dialogs are modal on construction.

Fix: NullPointerException in SearchableComboBox.java

Description

In displaying a string representation of items in the SearchableComboBox, toString() was potentially called on null items, resulting in a NullPointerException.

Cause

Failure to check for null items in the SearchableComboBox before calling toString() on those items.

Fix

Added checks for null data items, such that toString() is only called on non-null data items.

Notes

Tested by instantiating the SearchableComboBox with null data items. No NullPointerExceptions were thrown. Non-null data items were displayed correctly.

Fix: Spelling error in the html explanation of the Brier Skill Score

Description

Spelling errors were identified in the html description file for the Brier Skill Score (bss.html), which is displayed in the table of verification metrics in VerificationB.java.

Cause

Spelling errors in the html description file.

Fix

Corrected the spelling errors.

Fix: Real-valued verification thresholds not aggregated properly

Description

When aggregating verification metrics by averaging across the inputs, the real-values associated with the probability thresholds were not previously aggregated. Rather, the nominal value of the output threshold was equal to the real value associated with the first input. For consistency with the aggregation procedure applied to the metric values, the same aggregation function is now applied to the input thresholds, and the output threshold comprises an aggregate of the input thresholds. This is a nominal value only, since the actual verification is done separately for each input.

Cause

Failure to aggregate the real-valued thresholds associated with an AU when conducting aggregation by averaging metrics.

Fix

Added a method, `aggregate()`, to `DoubleProcedureParameter.java`. The method conducts averaging of the real-valued thresholds associated with multiple VUs when forming an average of the inputs metrics. Updated the `aggregate()` method of `MetricResultByThreshold.java`, which controls the aggregation across multiple metrics, in order to call the new `aggregate()` method in `DoubleProcedureParameter.java`.

Notes

Examined the real-valued thresholds associated with the AUs after the fix, which were correctly averaged.

Fix: Incorrect re-scaling of aggregation weights when missing first metric

Description

Failure to correctly re-scale the weights associated with multiple VUs when aggregating across missing metrics and, specifically, when the first metric is missing. For example, when aggregating metrics across three basins {A,B,C} with weights of {1/3,1/3,1/3}, a missing metric for basin A {MISSING, B, C} resulted in an aggregation of $1/3*B+1/3*C$ instead of $1/2*B+1/2*C$.

Cause

Failure to rescale aggregation weights properly in the `aggregate()` method of the `MetricResultByThreshold.java` class when the first metric is missing.

Fix

Fixed the `aggregate()` method of the `MetricResultByThreshold.java` class to correctly rescale weights when the first of several metrics is missing.

Notes

Checked the rescaled weights by calling the `aggregate()` method of the `MetricResultByThreshold.java` class with a missing metric in the first element. The weights were correctly rescaled.

Fix: Changed default behavior for “reset” buttons

Description

Previously, the default behaviour for the “reset” buttons in the GUI dialog `evs.gui.windows.MoreInputDialog.java` was to clear all contents in the associated tables rather than reset the contents to their original state.

Cause

The default behavior of the “reset” buttons in the GUI dialog `evs.gui.windows.MoreInputDialog.java` was misleading, clearing the contents of the associated tables, rather than resetting the contents to the original state.

Fix

Updated the default behavior of the methods associated with the “reset” `evs.gui.windows.MoreInputDialog.java` to return the contents of the tables to their original state, rather than simply clearing contents.

Notes

Checked the contents of the table after executing the updated reset options. The updated reset performed as expected, returning the contents of the tables to the original state, rather than clearing the contents.

Enhancement: Renamed `MoreOutputOptions.java`

Description

The naming convention for EVS advanced dialogs is to append “dialog” to the class name. Thus, renamed `MoreOutputOptions.java` to `MoreOutputOptionsDialog.java` for consistency.

Cause

Inconsistent naming of `MoreOutputOptions.java`.

Fix

Renamed `MoreOutputOptions.java` to `MoreOutputOptionsDialog.java`. The refactoring was conducted safely.

Notes

The refactoring was conducted safely in the IDE. Tested by cleaning and building the EVS. No build errors were encountered.

Enhancement: Added functionality to allow for joint pairing

Description

When comparing verification metrics from two forecasting systems, whether informally or using an explicit measure of skill, it is generally preferred to use common forecast valid dates and times. In order to support this, a mechanism was implemented to jointly pair two VUs; that is, to select only the common pairs from both VUs. For example, if the pairs used in the denominator of a skill score cover a different period than the pairs used in the numerator, the skill may reflect a combination of data variability (not desirable) and real differences in forecast quality (desirable).

Cause

Previously, it was not possible to coordinate two VUs and conduct verification for only those pairs that appear in both VUs.

Fix

Changes were required to several EVS classes, including GUI and non-GUI classes. In particular, updated the `getMergedData()` method of `PairedData.java` to compute the intersection of two paired inputs while only returning the pairs associated with the first input (i.e. pairs that are also present in the second input). Also, renamed the `getMergedData()` methods to `getIntersection()`. Updated `VerificationUnit.java` to store the VU on which joint pairing is conducted, together with associated getter and setter methods. Also updated the `getConditionalPairs()` method of `VerificationUnit.java` to compute the conditional pairs subject to any newly defined constraint on joint pairing. Added a new method `syncJointPairs()` to `VerificationA.java` to synchronize any verification units that are used as joint pairs following a change in the name of the VU. Updated the read/write methods of `ProjectFileIO.java` to read and write the new parameter for joint pairing, which appear in the `<paired_data>` block of the EVS project file. Finally, updated the `MoreOutputOptionsDialog.java` to include a new row for specifying a VU on which to conduct joint pairing and updated the `deleteSelectedUnits()` method of `VerificationA.java` to warn on deleting a verification unit that is used elsewhere for joint pairing. Performed extensive testing of the new functionality.

Enhancement: Improved organization of `MoreParOptionsDialog.java`

Description

The `close()` method of `MoreParOptionsDialog.java` failed to make best use of existing code elsewhere in `MoreParOptionsDialog.java`. By splitting `close()` into two methods, `saveParameters()` and `close()`, code re-use is improved.

Cause

Failure of `close()` method to make best use of existing code.

Fix

Split the `close()` method into two separate methods, `saveParameters()` and `close()`.

Enhancement: Allow real-valued bounds on computing metrics

Description

Metrics may be computed with thresholds that are defined in real units or climatological probabilities. When defining thresholds in terms of climatological probabilities, the corresponding real values are not known in advance. It is sometimes convenient to specify real-valued bounds on the thresholds for which metrics are computed, avoiding wasteful computations.

Cause

Inability to constrain the real-valued thresholds for which metrics are computed (e.g. when using climatological probabilities whose real-valued thresholds are not specified in advance).

Fix

Added real-valued bounds to constrain the computation of metrics for thresholds that fall within those bounds only. This required additions to several source files, namely `DoubleProcedureArrayParameter.java`, `MetricParameter.java`, `ThresholdMetricStore.java`, `DoubleProcedureParameter.java`. It also required a new parameter to store the bounds, `DoubleIntervalParameter.java`. Source files associated with the GUI and reading/writing of EVS project files were also updated to accommodate the new parameters, namely `ProjectFileIO.java`, `VerificationB.java` and `MoreParOptionsDialog.java`.

Notes

Constructed a `ThresholdMetricStore.java` with real-valued bounds on the thresholds accepted. Conducted verification and checked that the outputs were constrained to the predefined thresholds.

Enhancement: Detection limit on forecasts and observations

Description

In forecasting and observing some variables (e.g. precipitation), a detection limit may apply, beyond which it may be preferred to assume a lower or upper bound. For example, precipitation forecasts falling below an instrument detection limit might be assigned zero. This may avoid spurious verification results or sensitivities to choice of threshold (such as the threshold for determining Probability of Precipitation).

Cause

Inability to assign detection limits to forecasts and/or observations prior to conducting verification.

Fix

Implemented a detection limit in a new class, `DetectionLimit.java`, comprising a bound, limit, and function type. Any value that meets the logical test defined by the function type with respect to the limit (e.g. less than the limit) is assigned the value of bound (e.g. 0.0). A detection limit is applied jointly to the forecasts and observations and is associated with a particular VU (for which a new getter and setter method was implemented). When defined, the detection limit is used by the `getConditionedPairs()` method of `VerificationUnit.java`. This method returns conditional pairs after imposing any detection limit on the forecasts and observations. If a change in measurement units is requested, the detection limit is defined in the **target** measurement units of the forecasts and observations (i.e. in the units of the conditional paired data). The detection limit is an “advanced” option, not available in the EVS GUI. Rather, it is specified in the EVS project file under the `<paired_data>` block using a new `<detection_limit>`. The `<detection_limit>` comprises a `<limit>`, `<bound>`, and `<type>`, where the latter may include: `isLess`, `isGreater`, `isLessEqual`, and `isGreaterEqual`.

Notes

Evaluated the conditional pairs returned by the `getConditionalPairs()` method of `VerificationUnit.java` both with and without a detection limit. The resulting pairs properly reflected the detection limits imposed.

Enhancement: Option to exit MoreInputDialog from the Forecast Scale table

Description

Previously, there was no option to exit the MoreInputDialog.java via an “OK” button from the Forecast Scale table. Rather, it was necessary to navigate to one of the adjacent tabbed panels of the MoreInputDialog.java to exit. While the previous behavior is expected usage on first entering data, it is inconvenient on editing data, since editing may focus on the Forecast Scale table only (from which an exit sequence via a call to “OK” should be possible).

Cause

Inability to exit the MoreInputDialog.java from the tabbed panel containing the Forecast Scale table.

Fix

Added an exit (“OK”) option to the tabbed panel of the MoreInputDialog.java containing the Forecast Scale table.

Notes

Tested the “OK” button following implementation. The button worked as expected, allowing the dialog to exit.

Enhancement: Separate pairing options from the “Other options” dialog.

Description

The MoreInputDialog.java comprises a tabbed panel with “Other options.” This tabbed panel collects together a broad range of options associated with the input data. Many of these options relate to pairing of forecasts and observations. In order to improve the organization of these options and, specifically, to separate out the pairing options, a new tabbed panel of “Pairing options” was required MoreInputDialog.java.

Cause

Lumping together of too many options in the “Other options” table of the MoreInputDialog.java, including multiple options that control the pairing of forecasts and observations.

Fix

Implemented a new “Pairing options” table in the MoreInputDialog.java and moved all options related to pairing from the “Other options” table to the new table of pairing options.

Notes

Tested all options available in both the “Other options” table and the new “Pairing options” table. All options worked as expected.

Enhancement: Improved control on aggregating forecasts and observations.

Description

Added new functionality to the EVS pairing process to allow for improved control on a change of scale of the forecasts and observations prior to conducting pairing. Previously, the choices for aggregation (and the associated controlling variables) were shared between methods used to aggregate forecasts and observations *prior* to pairing and methods used to aggregate the pairs themselves. There was no separate control on aggregating forecasts and observations before pairing. This prevented pairing in some cases (i.e. when different parameter values were required for aggregation before pairing versus aggregation after pairing). For example, if the observed data begin at 12 UTC and comprise 6-hourly values, while the forecasts begin at 0 UTC and comprise daily values, separate control on the observations would allow the first two observations to be skipped and pairing to proceed at the aggregated daily scale. Subsequently, the pairs may be aggregated to explore verification results at the 5-daily scale.

Cause

Linking of functionality for aggregating forecast and observations *before* pairing to aggregating of pairs, i.e. shared variable values for these two scenarios. This prevented separate control of aggregation for pairing purposes, which is important when the forecasts and observations are defined at different scales (i.e. when aggregation of one or the other is essential prior to pairing).

Fix

Added new variables to `evs.analysisunits.VerificationUnit`, namely the `fcstAggStartHourUTC` and the `obsAggStartHourUTC`, together with the `fcstStartLeadHourUTC` to control, respectively, the start time for aggregating observations, the start time for aggregating forecasts, and the first forecast lead time at which to begin aggregation. Implemented corresponding getter and setter methods. Updated the EVS project file I/O to accommodate the new variables. Added a new table to the advanced options for input data, `evs.gui.windows.MoreInputDialog.java`, to provide access to the new options. The new table comprises these and other options for pairing data.

Notes

Tested the new functionality by using forecasts and observations that required separate aggregation before pairing, together with aggregation after pairing. The separate options worked as expected.