06 Lookup Table Module - EOL in 2023

Lookup Table module configuration

The Lookup table module is used to derive a simple value based on combining input values of different time series in the forecast database. These are then used to search in a multi-dimensional lookup table to derive the requested output. The module may also be employed to derive a decision based on a hierarchic set of rules (critical conditions table).

The lookup table utility is predominantly applied as the forecasting tool for coastal forecasting. Typically values such as predicted surge, wind force and direction, wave height, fluvial flow in an estuary are used to predict values at a number of points on the coast or in an estuary. These values are generally defined as a Lookup Index. This can then be resolved to a text string such as "Flood Warning" or "Severe Flood Warning" for use in for example reports using the ValueAttributeMaps (see Regional Configuration).

Three main types of lookup table may be defined;

- simple table lookup. This is a two column (or row) table where the value at each time step in the input series is used to identify a relative position in the first column (or row). The result value is found in the second column (or row) at the same relative position.
- Multi-dimensional lookup. This is a lookup in a matrix. Two input series are required. One is used to find the relative row position in the matrix at each time step, while the other is used to find the relative column position in the matrix. The output value is found through resolving these relative positions in the matrix values using bi-linear interpolation.
- Critical condition tables. These defined a set of heuristic rules. Multiple inputs can be combined and an output is found through evaluating the heuristic rules. A default output (also using rules can be defined).

When available as configuration on the file system, the name of the XML file for configuring an instance of the general adapter module called for example Coastal_Lookup_Forecast may be:

Coastal_Lookup_Forecast 1.00 default.xml

Coastal_Lookup_Forecast	File name for the Coastal_Lookup_Forecast configuration.
1.00	Version number
default	Flag to indicate the version is the default configuration (otherwise omitted).

The lookup table module can be included both in a normal workflow run on the Forecasting Shell Server, or in a workflow to be run interactively through the lookup table display (See display configuration). Configuration for use in either is the same. The only difference is that in the latter case the result data is not available for viewing outside the lookup table display, and reports are generated for viewing locally only.

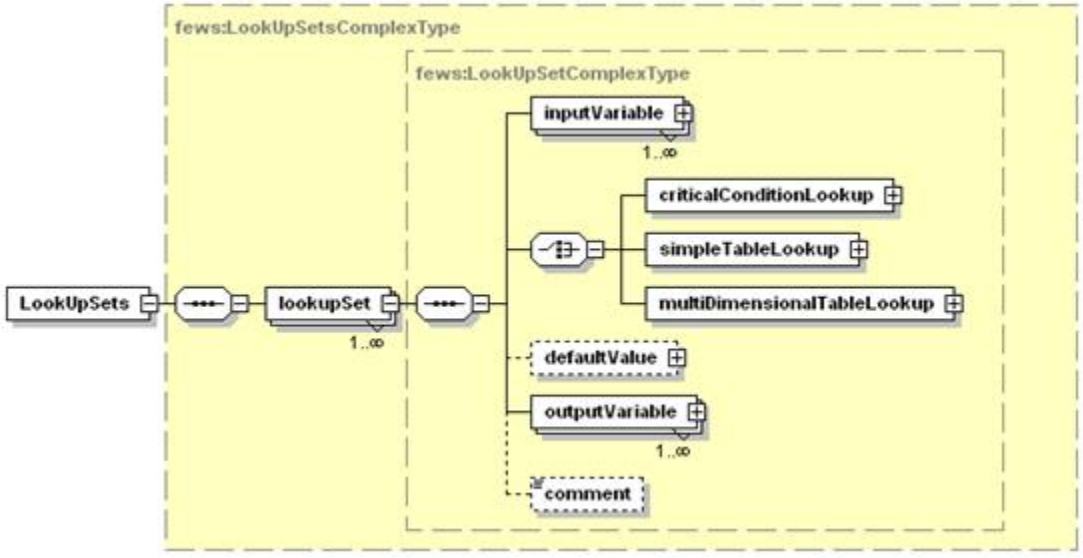

Figure 78 Elements of the lookup table configuration.

LookupSet

Root element of the definition of a lookup table. Multiple entries may exist.

Attribute;

- **lookupSetId** : Id of the lookup table. Used for reference purposes only (e.g. in log messages).

inputVariable
Definition of input variable to be used in the lookup table. For each entry in the lookup table an input variable will need to be identified. The variableId is used to refer to the time series. See Transformation Module for definition of inputVariable configuration.

outputVariable
Definition of output variable as a result of the lookup table. A single timeSeriesSet for one location output variable per lookup table is defined.

comment
Optional comment on lookup display configuration. Used for reference purposes only.

criticalConditionLookup
Root element for definition of a critical condition table. If no results can possibly be returned by any of the conditions specified, a defaultValue should be defined as a set of rules.

Attributes;

- **Id** Id of the criticalConditionTable. Used for reference purposes only

simpleTableLookup
Root element for definition of a simple table lookup. Multiple entries may exist.

Attributes;

- **lookUpVariableId** Id of the input variable to be used in the table.
- **outputVariableId** Id of the output variable.
- **rows** number of rows in lookup table (if lookup is defined per row then this is equal to 1).
- **cols** number of columns in lookup table (if lookup table is defined per col then this is equal to 1).
- **type** Optional indication of type of value in lookup table (same type will be returned). Enumeration of "float"or "int".

multiDimensionalLookup
Root element for definition of a multidimensional lookup table. Multiple entries may exist.

Attributes;

- **lookUpRowVariableId** Id of the input variable to be used in the table for finding relative row position.
- **lookUpColVariableId** Id of the input variable to be used in the table for finding relative column position..
- **outputVariableId** Id of the output variable.
- **rows** number of rows in lookup table (matrix).
- **cols** number of columns in lookup table (matrix).
- **type** Optional indication of type of value in lookup table. Enumeration of "float"or "int".

criticalConditionLookup

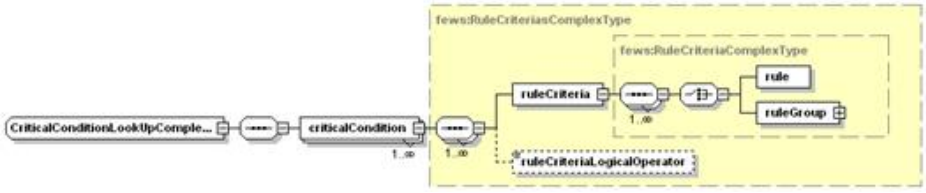

Figure 79 Elements of the criticalConditionLookup configuration

criticalCondition
Definition of a critical condition as a set of rules. Multiple entries may exist. When multiple entries do exist, then these will be resolved sequentially until a condition defined is met. The result is then written to the output time series. Each condition holds a set of rules. Each rule is resolved to a Boolean true or false. Rules can be combined in ruleGroups using Boolean operators. If a "true" value returned through combination of all rules and ruleGroups specified, then the conditions specified are met.

Attributes (only required attributes defined);

- **rule:** string value for result to be returned if conditions specified are met (for reference purposes only).
- **ruleIndex:** index value returned if conditions specified are met. This is the value returned in the output time series. Value given is either a numerical value enclosed in quotes (e.g. "4" or "Missing" to indicate a missing value should be returned).

ruleCriteria
Root element for definition of set of rules and ruleGroups. Multiple ruleCriteria can be defined. These are combined using the logical operator defined.

ruleCriteriaLogicalOperator

Operator for combining ruleCriteria to single Boolean value. Enumeration of "*and* " and "*or* ".

rule

Definition of a rule to resolve to a Boolean value.

Attributes;

- variable: id of Input variable to use in evaluating rule
- operator: Definition of operator to be used in comparison. Enumeration of options include;
- *lt* : less than
- *le* : less than or equal to
- *eq* : equal to
- *ge* : greater than or equal to
- *gt* : greater than
- *ne* : not equal to
- value: Value to compare input variable to using operator defined.
- logical: optional definition of logical operator to combine sequence of rules (for rules defined in a rule group only). Enumeration of "*and* " and "*or* ".

ruleGroup

Root element for defining a rule group. A rule group is a sequence of rules. Each rule is configured as defined above, and combined using the logical operator given in the rule. The logical operator need not be included in the last rule defined.

Example:

```
<!--All following conditions occur when wind force >= 4 -->
<!--Condition for Wind N & NE (>= 0 and < 67.5 degrees) -->
<criticalCondition rule="Severe Flood Warning" ruleIndex="1">
   <ruleCriteria>
      <rule variable="WindDir" operator="ge" value="0" logical="and"/>
      <rule variable="WindDir" operator="lt" value="67.5"/>
   </ruleCriteria>
   <ruleCriteriaLogicalOperator>and</ruleCriteriaLogicalOperator>
   <ruleCriteria>
      <ruleGroup>
         <rule variable="WindForce" operator="ge" value="10"
                                                  logical="and"/>
         <rule variable="TotalWaterLevel" operator="ge" value="5.60"
                                                  logical="or"/>
      </ruleGroup>
      <ruleGroup>
         <rule variable="WindForce" operator="ge" value="4"
                                                  logical="and"/>
         <rule variable="TotalWaterLevel" operator="ge" value="5.90"/>
      </ruleGroup>
   </ruleCriteria>
</criticalCondition>
```

defaultValue

The default value element is identical to the specification of a criticalConditon as described above.

SimpleTableLookup

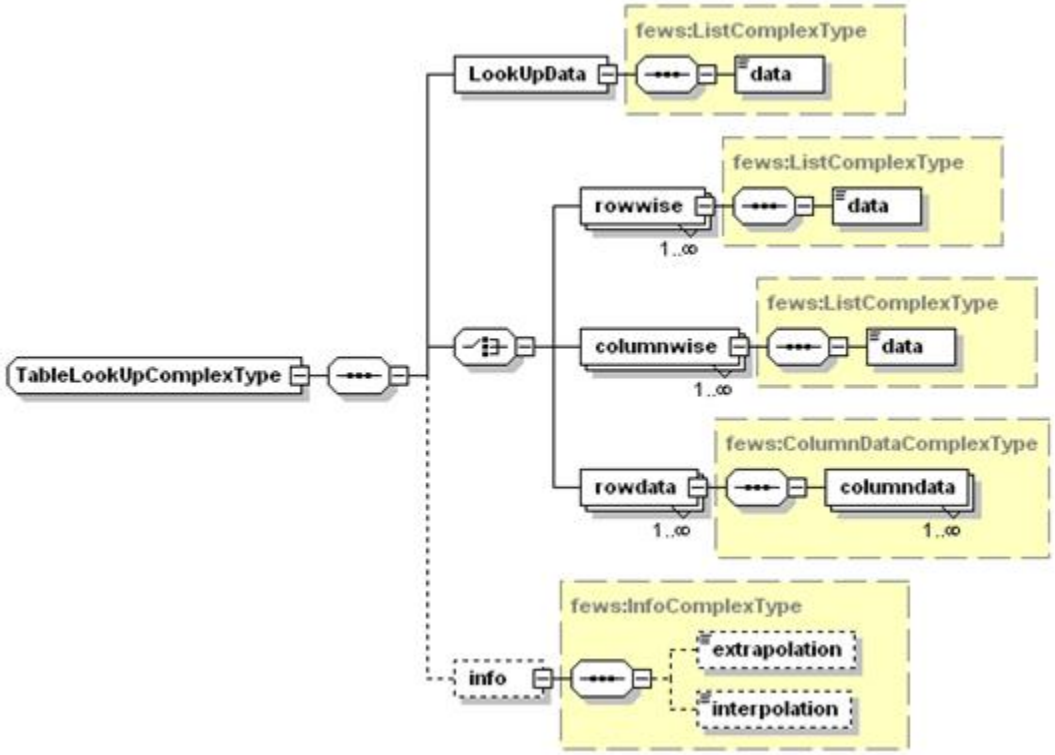

Figure 80 Elements of the simpleTableLookup configuration

LookUpData

Row vector of data used to find relative position of input variable.

Attributes;

- **number** : optional definition of number of entries (otherwise inferred from data provided)
- **type** : optional type indication of data. Enumeration of "float" and "int".
- **separator** : optional indication of separator string used between values. Default is space. Enumeration of;
- *space*

data

Element containing data vector separated by separator character defined.

rowwise

Element to define vector of data to lookup data in. Data value at relative position is returned. Attributes are same as LookUpData element. Use this element if data is provided as a row.

columnwise

Element to define vector of data to lookup data in. Data value at relative position is returned. Use this element if data is provided as a one value per row (as a column)

Attributes;

- **number** : optional definition of number of entries (otherwise inferred from data provided)
- **type** : optional type indication of data. Enumeration of "float" and "int".
- **separator** : optional indication of separator string used between values. Default is space. Enumeration of;
- *lineseparator*

info

Element containing information on how values are determined in lookup vector using the relative position determined.

info:extrapolation

Definition of how to extrapolate when relative position is above last or below first value in vector. Enumeration includes;

- *none* : no extrapolation, missing value is returned
- *minmax* : limit values returned to minimum/maximum of vector
- *linear* : linear extrapolation using last or first two values in vector

info:interpolation

Definition of how to interpolate between values in vector. Enumeration includes;

- *class* : returns closest value in vector.

- *linear* : linear interpolation

Example:

```
<simpleTableLookup lookUpVariableId="X1" outputVariableId="Y1" rows="1"
                                          columns="5" type="float">
   <LookUpData type="float" seperator="space">
      <data>17.80    18.90    19.21    19.51    19.82</data>
   </LookUpData>
   <rowwise number="1" type="float" seperator="space">
      <data>3.65    5.67    6.10    6.40    6.74</data>
   </rowwise>
   <info>
      <interpolation>linear</interpolation>
   </info>
</simpleTableLookup>
```

MultipeDimensionLookup

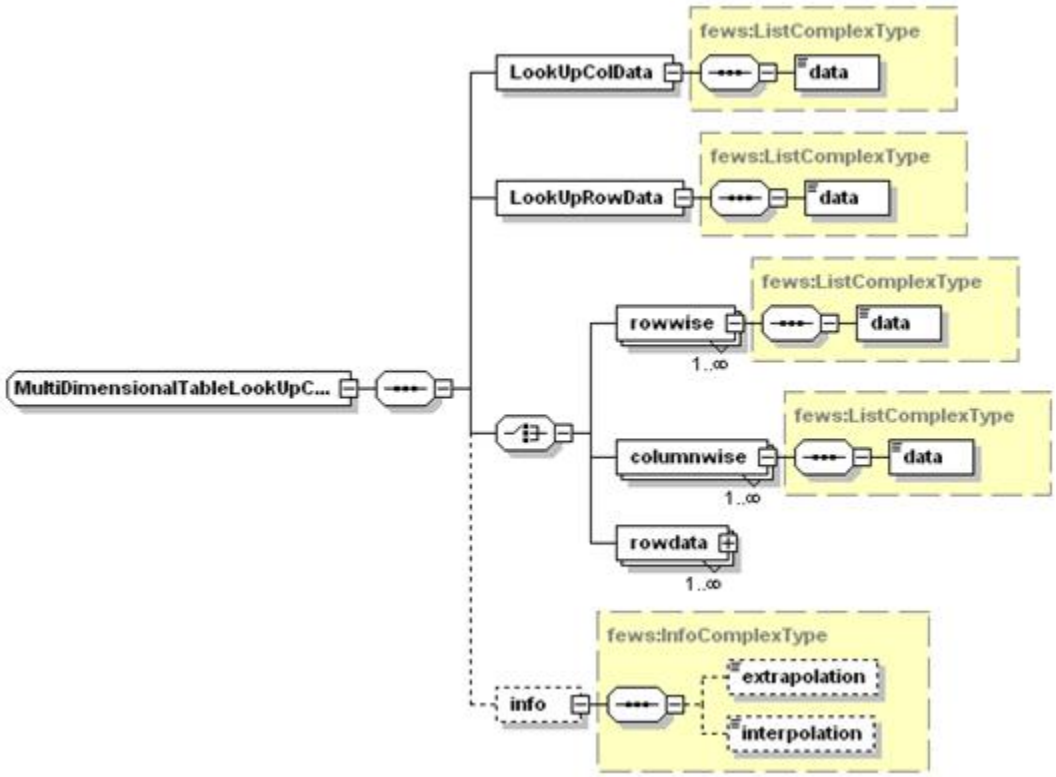

Figure 81 Elements of the multiDimensionalLookup configuration

lookupColData

Row vector of data used to find relative position in matrix columns of input variable defined as lookUpColVariableId.

Attributes;

- **number** : optional definition of number of entries (otherwise inferred from data provided)
- **type** : optional type indication of data. Enumeration of "float" and "int".
- **separator** : optional indication of separator string used between values. Default is space. Enumeration of;
- *space*

lookupRowData

Row vector of data used to find relative position in matrix rows of input variable defined as lookUpRowVariableId.

Attributes;

- **number** : optional definition of number of entries (otherwise inferred from data provided)
- **type** : optional type indication of data. Enumeration of "float" and "int".

- **separator** : optional indication of separator string used between values. Default is space. Enumeration of;
- *space*

rowwise

Element for defining rows of marix as a vector of data on one line. For definition see simpleTableLookup. The number of rowwise elements provided must be equal to the number of columns defined in the multiDimensionalLookup element. Each rowwise vector must contain as many values as defined in cols in the multiDimensionalLookup element.

colwise

Element for defining rows of marix as a vector of data on one multiple lines. For definition see simpleTableLookup element. The number of colwise elements provided must be equal to the number of columns defined in the multiDimensionalLookup element. Each colwise vector must contain as many values as defined in cols in the multiDimensionalLookup element.

Info

See definition in simpleTableLookup element

Example:

```
<multiDimensionalTableLookup lookUpColVariableId="KnaithPLWL"
                             lookUpRowVariableId="KeadbyPHWL"
                             outputVariableId="KnaithPHWL"
                            rows="3" columns="5" type="float">
   <LookUpColData type="float" seperator="space">
      <data>1.67    3.80    4.90</data>
   </LookUpColData>
   <LookUpRowData type="float" seperator="space">
      <data>3.70    4.10    4.60    5.30    5.80  </data>
   </LookUpRowData>
   <rowwise number="1" type="float" seperator="space">
      <data>2.56    2.78    3.22    3.50    3.61    </data>
   </rowwise>
   <rowwise number="2" type="float" seperator="space">
      <data>4.65    4.90    5.30    5.70    5.90    </data>
   </rowwise>
   <rowwise number="3" type="float" seperator="space">
      <data>5.30    5.50    5.70    6.10    6.35    </data>
   </rowwise>
   <info>
      <interpolation>linear</interpolation>
   </info>
</multiDimensionalTableLookup>
```