

Aggregation Accumulative

- [Accumulative](#)
 - [Input](#)
 - [Options](#)
 - [Output](#)
 - [Description](#)
 - [Validation rules](#)
 - [Configuration example](#)

Accumulative

Input

- **inputVariable.** (Be careful with using non equidistant input data, this transformation is not designed for that and could give unexpected output values. In case of non-equidistant data the [Statistics Serial Sum Transformation](#) might be more suitable)

Options

- **validationRules.** It is possible to define a set of validationRules. With these validation rules it is possible to define a set of criteria which determines the outputflag of the calculated value based on the number of missing values and/or doubtful values counted in the input values.

Output

- **outputVariable.**
Please note that the timestep of the output variable needs to have the same "hour" value. E.g. aggregation from daily to monthly timestep works when both are defined for 00:00 GMT, but not when one is set of 08:00 GMT and the other for 00:00 GMT. This typically is only an issue with user-defined timesteps. The default timesteps of Delft-FEWS (or "units") for day, week and year all have a timestamp at midnight.

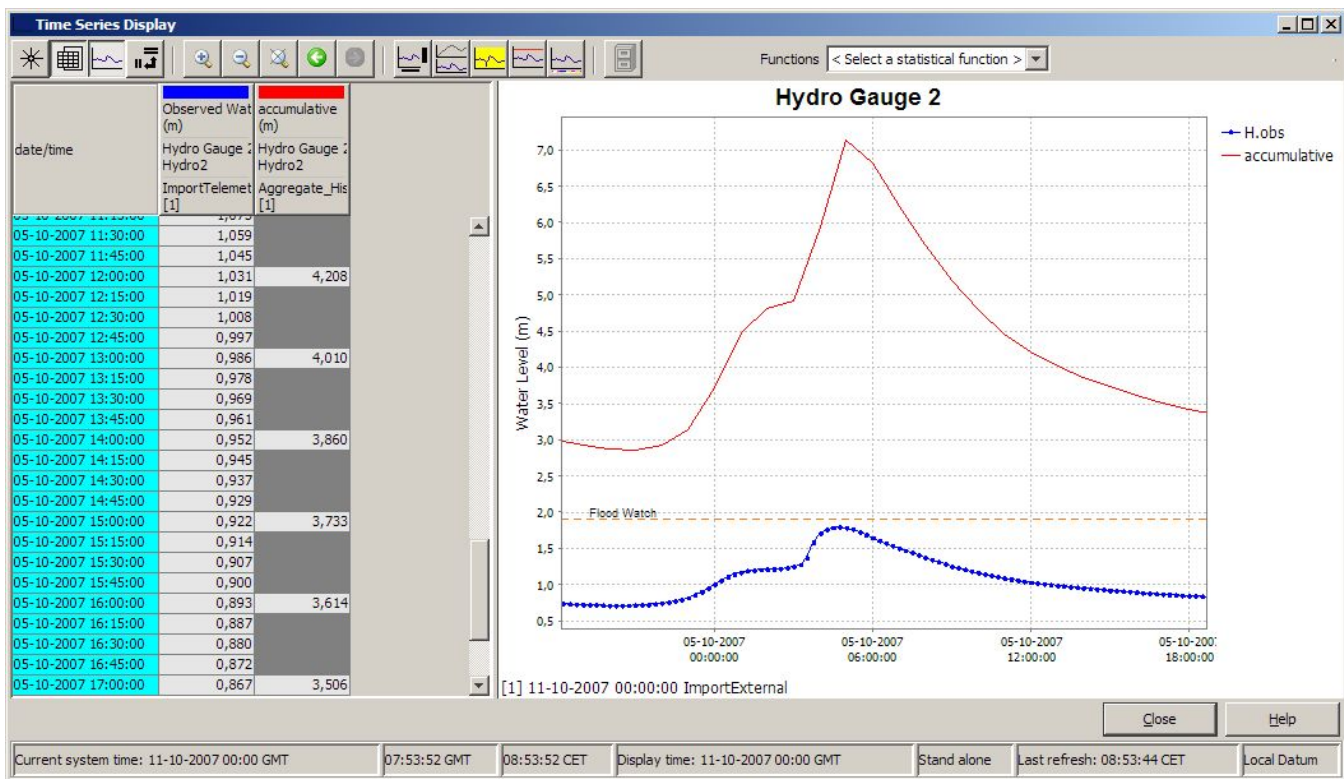
Description

This transformation performs an aggregation from an instantaneous time series to an aggregated time series. This procedure sums the values of the input timeseries that are within the aggregation period. If no aggregation period is configured, then the aggregation period is equal to the period between the current output time and the previous output time. Alternatively the aggregation period can be configured in the time series set of the output variable. In that case the aggregation period is relative to the current output time and aggregation periods for different output times are allowed to overlap. Using overlapping aggregation periods it is possible to use this transformation to calculate a moving sum. If one of the input values is missing or unreliable the output is missing.

The table below shows an example of accumulating 6-hourly values to daily values using this transformation.

	Original series	Result
Date/Time	Value	Value
01-01-2007 00:00	1.00	
01-01-2007 06:00	2.00	
01-01-2007 12:00	3.00	
01-01-2007 18:00	4.00	
02-01-2007 00:00	5.00	14.00
02-01-2007 06:00	6.00	
02-01-2007 12:00	NaN	
02-01-2007 18:00	8.00	
03-01-2007 00:00	9.00	NaN
03-01-2007 06:00	10.00	

The figure below shows original 15 minute data and the aggregated hourly data using the accumulative function:



Validation rules

See this wiki [Validationrules](#) for a general explanation, and below for a config example including validationrules.

Configuration example

Below a configuration example which has proven itself in an operational context.

The idea behind it is that if the majority of the input is doubtful, the output should be doubtful and if the majority of the input is missing/unreliable the output should be missing.

Splitting into quartiles turned out to be most intuitive, distinctive and least complex.

The outputCustomFlagSourceId follows the naming convention of the amount of doubtful and missing/unreliable quartiles. For example "d3u1" stands for 75% doubtful and 25% missing/unreliable. This way the reason for the quality flag is easy to understand when shown in FEWS.

Conditional Aggregation used at Dutch Waterboard HDSR

```
<?xml version="1.0" encoding="UTF-8"?>
<transformation id="DebietBerekeningDag">
  <aggregation>
    <accumulative>
      <inputVariable>
        <timeSeriesSet>
          <moduleInstanceId>DebietBerekening</moduleInstanceId>
          <valueType>scalar</valueType>
          <parameterId>Q.berekend.15min</parameterId>
          <locationSetId>HHNK_TMX_debiet</locationSetId>
          <timeSeriesType>external historical</timeSeriesType>
          <timeStep unit="minute" multiplier="15"/>
          <relativeViewPeriod unit="day" start="-300" end="0"/>
          <readWriteMode>read only</readWriteMode>
          <synchLevel>1</synchLevel>
        </timeSeriesSet>
      </inputVariable>
      <validationRule>
        <inputDoubtfulPercentage>0</inputDoubtfulPercentage>
        <inputMissingPercentage>0</inputMissingPercentage>
        <outputValueFlag>reliable</outputValueFlag>
      </validationRule>
    </accumulative>
  </aggregation>
</transformation>
```

[illegible]

```

        <outputValueFlag>missing</outputValueFlag>
        <outputCustomFlagSourceId>d0u3</outputCustomFlagSourceId>
    </validationRule>
    <validationRule>
        <inputDoubtfulPercentage>100</inputDoubtfulPercentage>
        <inputMissingPercentage>75</inputMissingPercentage>
        <outputValueFlag>missing</outputValueFlag>
        <outputCustomFlagSourceId>dlu3</outputCustomFlagSourceId>
    </validationRule>
    <validationRule>
        <inputDoubtfulPercentage>100</inputDoubtfulPercentage>
        <inputMissingPercentage>100</inputMissingPercentage>
        <outputValueFlag>missing</outputValueFlag>
        <outputCustomFlagSourceId>d0u4</outputCustomFlagSourceId>
    </validationRule>
    <outputVariable>
        <timeSeriesSet>
            <moduleInstanceId>DebietBerekening</moduleInstanceId>
            <valueType>scalar</valueType>
            <parameterId>Q.berekend.15min</parameterId>
            <locationSetId>HHNK_TMX_debiet</locationSetId>
            <timeSeriesType>external historical</timeSeriesType>
            <timeStep unit="minute" multiplier="15"/>
            <relativeViewPeriod unit="day" start="-300" end="0"/>
            <readWriteMode>read only</readWriteMode>
            <synchLevel>1</synchLevel>
        </timeSeriesSet>
    </outputVariable>
</accumulative>
</aggregation>
</transformation>

```