Deltares
Enabling Delta Life

Delft-FEWS
Advanced Configuration Course (Continued)

Day 4 morning

- Timesteps and aggregation
- Performance indicators
- Manipulating grids (and building models) using embedded raster GIS: pcraster
- Client server issues

Time Series and Time Steps

Deltares

Time Series Sets – Again!

Properties of time series:

- location Id – or – locationSetId
- parameter Id
- timestep
- relativeViewPeriod

- valueType (scalar, grid, profile, ..)
- timeSeriesType (external historical, external forecast, ..)
- moduleInstanceId (import, PDM, KW, ISIS, ..)
- readWriteMode

- expiryTime (when to delete from database)
- synchLevel

- simple transformations (delay, multiplier, divider, incrementer)
- ensembleId

Deltares

Time Steps (1)

Time steps in FEWS

- Unit (second, minute, hour, day, week, month, year, nonequidistant)
- Multiplier (1, 2, ..)
- Divider (1, 2, ..)

Examples:

- <timeStep unit="hour"/>
- <timeStep unit="hour" multiplier="1"/>
- <timeStep unit="minute" multiplier="15"/>
- <timeStep unit="hour" divider="4"/>
- <timeStep unit="day" multiplier="1"/>

Not very complex!

Deltares

Time Steps (2)

Time steps in FEWS

- Unit (second, minute, hour, day, week, month, year, nonequidistant)
- Multiplier (1, 2, ..)
- Divider (1, 2, ..)
- timeZone (GMT, CET, WET, GMT+1, …)

Examples:

this not equal to this (different time series)

- <timeStep unit="day" multiplier="1"/>
- <timeStep unit="day" multiplier="1" timeZone="GMT"/>
- <timeStep unit="day" multiplier="1" timeZone="GMT+1"/>

- <timeStep unit="day" multiplier="1" timeZone="GMT+8"/>

Deltares

Time Steps (3)

Time steps in FEWS

- Unit (second, minute, hour, day, week, month, year, nonequidistant)
- Multiplier (1, 2, ..)
- Divider (1, 2, ..)
- timeZone (GMT, CET, WET, GMT+1, …)
- monthDays, times, daysOfMonth

Examples:

- meteorological seasons: <timeStep monthDays="--03-01 --06-01 --09-01 --12-01"/>
- every day at 13:00: <timeStep times="13:00"/>
- every day at 13:00 en 20:00: <timeStep times="13:00 20:00"/>
- every 12th of the month: <timeStep daysOfMonth="12"/>
- decade: <timeStep daysOfMonth="1 11 21"/>

Keep it simple when possible!

Deltares

Time Steps (4)

For the FEWS database a time series with a time step

<timeStep unit="day" multiplier="1" timeZone="GMT+8"/>

is different from a time series with a time step

<timeStep unit="day" multiplier="1" timeZone="GMT"/>

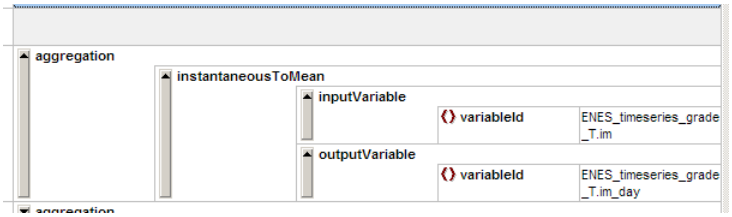
Deltares

Time Steps: Aggregation

The FEWS Transformation module can aggregate data to larger time steps

- Input: inputVariable with smaller timestep
- Output: ouputVariable with larger timestep

- accumulative
- instantaneous
- instantaneousToMean
- meanToMean



Deltares

Aggregation methods

accumulative – Sums the values of the input timeseries that fall within the output interval

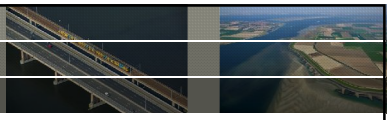
instantaneous – Sets the output value to the exact same value in the input timeseries at time t

instantaneousToMean – calculates the mean of the values of the input timeseries and stores the result in the output timeseries (two methods avaialble)

MeanToMean – similar to instantaneousToMean

Deltares

Accumulative

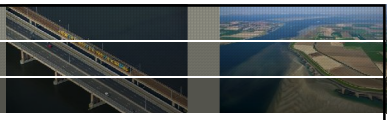


- This transformation performs an aggregation from an instantaneous time series to an aggregated time series.
- This procedure sums the values of the input timeseries that fall within the output interval. If one of the input values is missing or unreliable the output is missing.
- The table below shows an example of accumulating 6-hourly values to daily values using this method.

	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	

Deltares

Instantaneous



- This transformation performs an aggregation from an instantaneous input time series to an instantaneous output time series.
- Sets the output value to the exact same value in the input timeseries at time t. It simply samples points.
- The table below shows an example of accumulating 6-hourly values to daily values using this method.
- ***Output volume not necessary equal to input volume***

	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	5,00
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	9,00
03-01-2007 06:00	10,00	

Deltares

InstantaneousToMean

- This transformations calculates the mean value of instantaneous values over a certain period. If the options allowMissingValues is enabled a mean value is calculated if there 1 or more valid values in the aggregation period.
- If the option is disabled a missing value returned if one of the values in the period is a missing value.
- The transformation offers two options for calcuating the mean value over a period.
- The default method takes the mean of the last n pairs, averages that, and stores it at the output time.
- An alternate method (similar to the MeanToMean aggregation) is enabled by setting the includeFirstValueOfAggregationPeriodInCalculation option to false) takes the last n values that fit in the output interval (includding the time itself) and stores these at the output time

Deltares

InstantaneousToMean

Original series		Aggregated series					
	method->	*Alternate (like meanToMean)*		*Default*		*Alternate (like meanToMean)*	*Default*
	option->	*allow missing*	*allow missing*		*do not allow missing*	*do not allow missing*	
Date/Time	Value	Value	Calculation	Value	Calculation	Value	Value
01-01-2007 00:00	1						
01-01-2007 06:00	2						
01-01-2007 12:00	3						
01-01-2007 18:00	4						
02-01-2007 00:00	5	3,50	(2 + 3 + 4 + 5)/4	3	(((1+2)/2) + ((2+3)/2) + ((3+4)/2) + ((4+5)/2))/4	3,5	3
02-01-2007 06:00	6						
02-01-2007 12:00	7						
02-01-2007 18:00	NaN						
03-01-2007 00:00	9	7,33	(6 + 7 + 9)/3	6	(((5+6)/2)+((6+7)/2))/2	-	-
03-01-2007 06:00	10						

Deltares

MeanToMean

This transformation performs an aggregation from an mean input time series to an mean output time series.

The average value of the mean value in the aggregation period will be the calculated mean value for the output time series.

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

This method will give the same results as the instantaneousToMean transformation However, it has no option to ignore missing values in the input series.

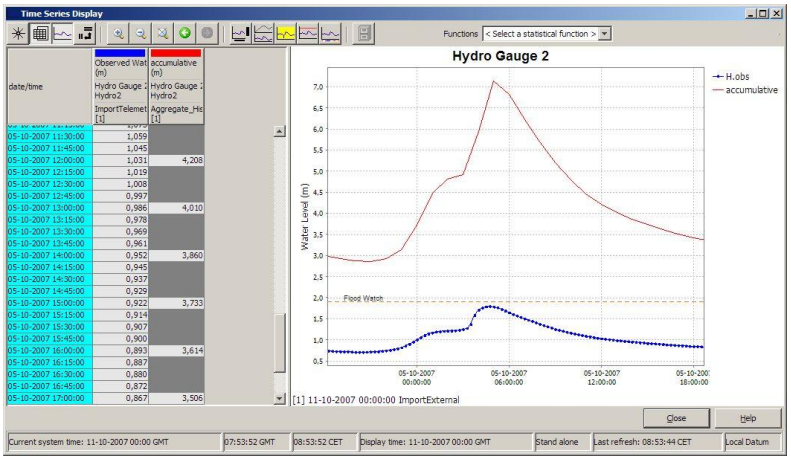
Output volume equal to input volume (if no missing values)

	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	3,5
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	

Deltares

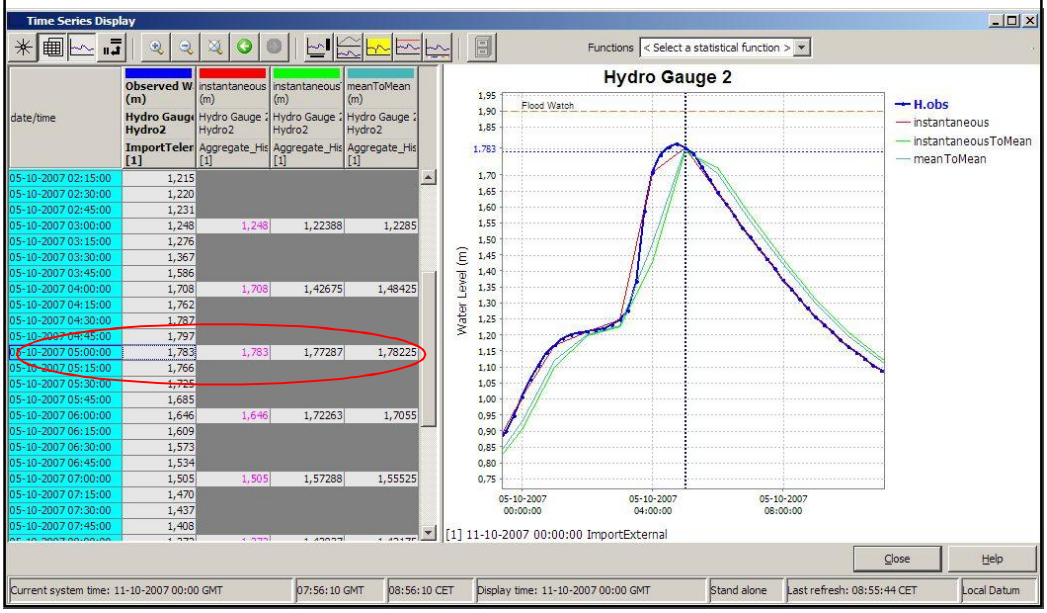
Aggregation: Examples

Aggregate rainfall from 15 min to 1 hour using **accumulative** function



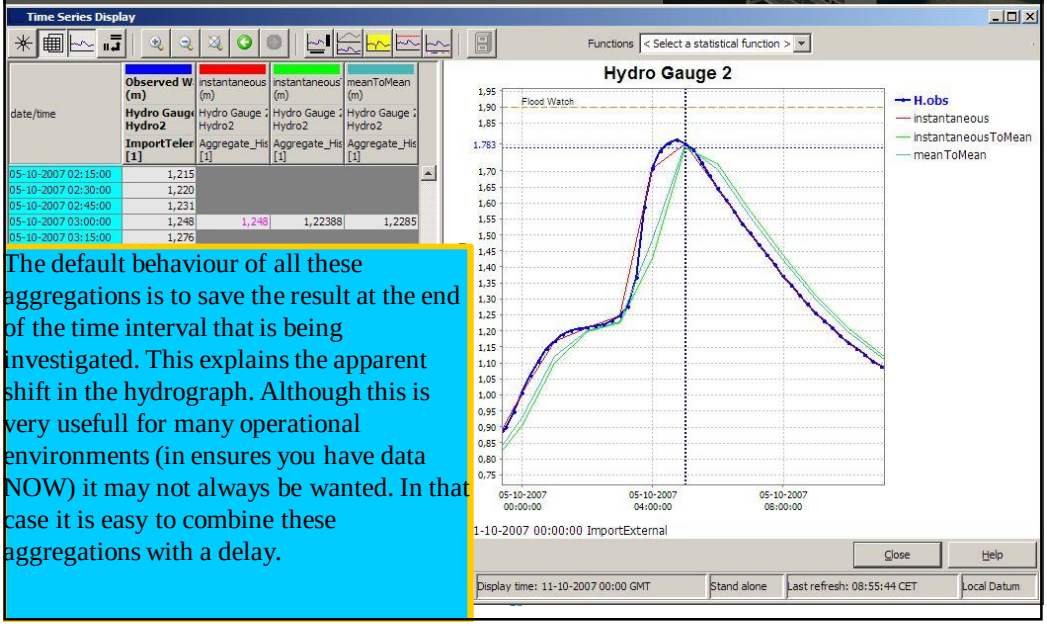
Deltares

Aggregation: Examples, Three methods….



date/time	Observed W (m) Hydro Gauge Hydro2 ImportTeler [1]	instantaneous (m) Hydro Gauge 2 Hydro2 Aggregate_His [1]	instantaneous (m) Hydro Gauge 2 Hydro2 Aggregate_His [1]	meanToMean (m) Hydro Gauge 2 Hydro2 Aggregate_His [1]
05-10-2007 02:15:00	1,215			
05-10-2007 02:30:00	1,220			
05-10-2007 02:45:00	1,231			
05-10-2007 03:00:00	1,248	1,248	1,22388	1,2285
05-10-2007 03:15:00	1,276			
05-10-2007 03:30:00	1,367			
05-10-2007 03:45:00	1,586			
05-10-2007 04:00:00	1,708	1,708	1,42675	1,48425
05-10-2007 04:15:00	1,762			
05-10-2007 04:30:00	1,787			
05-10-2007 04:45:00	1,797			
05-10-2007 05:00:00	1,783	1,783	1,77287	1,78225
05-10-2007 05:15:00	1,766			
05-10-2007 05:30:00	1,725			
05-10-2007 05:45:00	1,685			
05-10-2007 06:00:00	1,646	1,646	1,72263	1,7055
05-10-2007 06:15:00	1,609			
05-10-2007 06:30:00	1,573			
05-10-2007 06:45:00	1,534			
05-10-2007 07:00:00	1,505	1,505	1,57288	1,55525
05-10-2007 07:15:00	1,470			
05-10-2007 07:30:00	1,437			
05-10-2007 07:45:00	1,408			

Aggregation: Examples



date/time	Observed W (m) Hydro Gauge Hydro2 ImportTeler [1]	instantaneous (m) Hydro Gauge 2 Hydro2 Aggregate_His [1]	instantaneous (m) Hydro Gauge 2 Hydro2 Aggregate_His [1]	meanToMean (m) Hydro Gauge 2 Hydro2 Aggregate_His [1]
05-10-2007 02:15:00	1,215			
05-10-2007 02:30:00	1,220			
05-10-2007 02:45:00	1,231			
05-10-2007 03:00:00	1,248	1,248	1,22388	1,2285
05-10-2007 03:15:00	1,276			

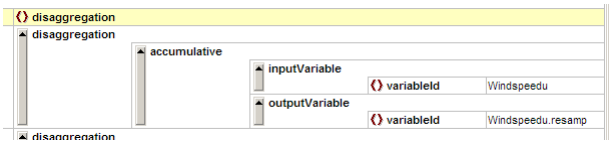
The default behaviour of all these aggregations is to save the result at the end of the time interval that is being investigated. This explains the apparent shift in the hydrograph. Although this is very usefull for many operational environments (in ensures you have data NOW) it may not always be wanted. In that case it is easy to combine these aggregations with a delay.

Time Steps: Dis-Aggregation

The FEWS Transformation module can dis-aggregate data to smaller time steps

- Input: timeSeriesSet with original timestep
- Output: timeSeriesSet with smaller timestep

- accumulative
- instantaneous
- meanToInstantaneous
- meanToMean
- weights



Deltares

DisAggregation methods

- accumulative – Divides the values of the input timeseries by the number of timesteps in the output timeseries and stores the resulting values at each step.
- instantaneous – Sets the output value to the exact same value in the input timeseries at time t
- MeanToInstantaneous – calculates the mean of the values of the input timeseries and stores the result in the output timeseries (two methods avaialble)
- MeanToMean – similar to instantaneousToMean
- weights – Similar to accumulative but a weight can be set for each output point. For example used to disaggregate daily ET to hourly values.

Deltares

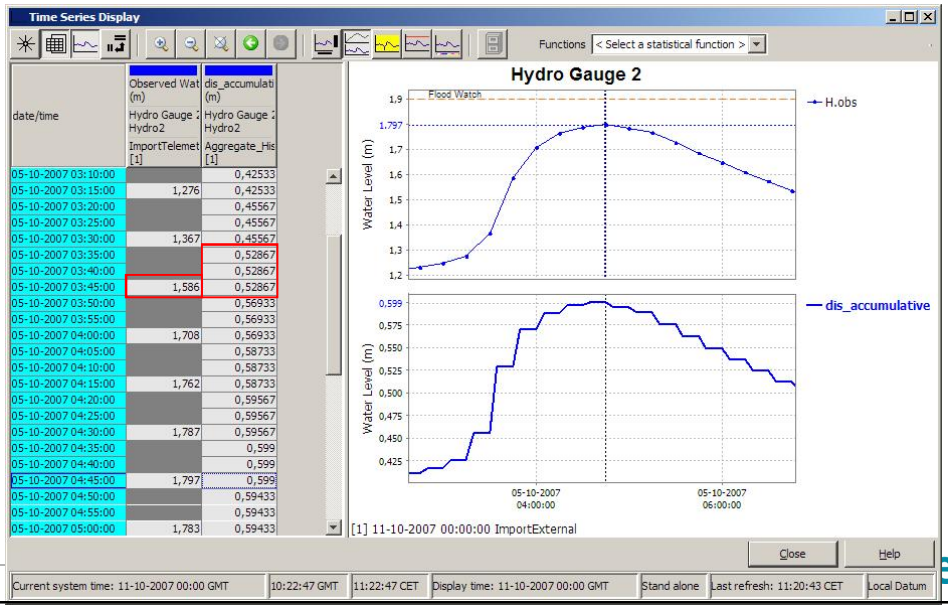
Accumulative

- This transformation performs a disaggregation on an accumulative input time series.
- Divides the values of the input timeseries by the number of timesteps in the output timeseries and stores the resulting values at each step.

	Input	Output
Date/Time	Value	Value
01-01-2007 06:00		1,25
01-01-2007 12:00		1,25
01-01-2007 18:00		1,25
02-01-2007 00:00	5,00	1,25
02-01-2007 06:00		1,75
02-01-2007 12:00		1,75
02-01-2007 18:00		1,75
03-01-2007 00:00	7,00	1,75

Deltares

Accumulative



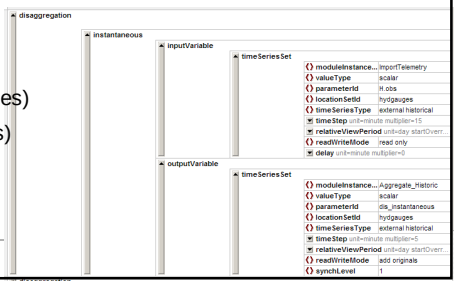
Instantaneous

- This transformation performs a disaggregation on an instantaneous input time series.
- The output values are copied from the input time series if a matching time exists in the input value.
- If this is not the case the output value is calculated by linear interpolation if the option interpolate is enabled.
- If this option is disabled the output value will be a missing value.

$Y = Y_0 + (Time_t - Time_0) * (Y_0 - Y_1)/(Time_0 - Time_1)$

in which:

- Y is the interpolation result in the output series
- Y_0 is the value of the first point before Y (in the input series)
- Y_1 is the value of the first point after Y (in the input series)
- Time_t is the date/time for Y (in the output series)
- Time_0 is first time before Time_t (in the input series)
- Time_1 is first time after Time_t (in the input series)



Instantaneous

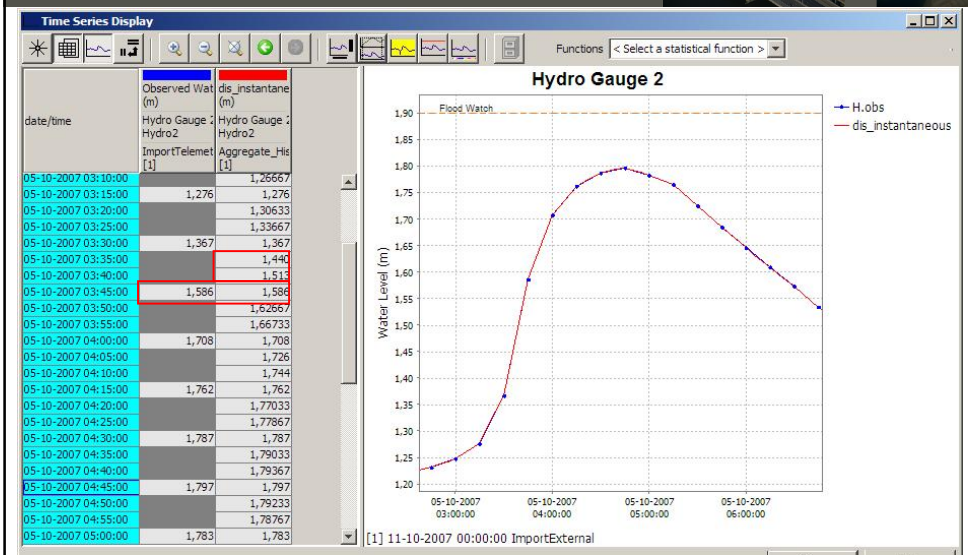
	Input	Output (interpolation)	Output (no interpolation)
Date/Time	Value	Value	Value
01-01-2007 00:00	10,00	10,00	10,00
01-01-2007 06:00		8,75	-
01-01-2007 12:00		7,50	-
01-01-2007 18:00		6,25	-
02-01-2007 00:00	5,00	5,00	5,00
02-01-2007 06:00		6,00	-
02-01-2007 12:00		7,00	-
02-01-2007 18:00		8,00	-
03-01-2007 00:00	9,00	9,00	9,00

Copied

Interpolated

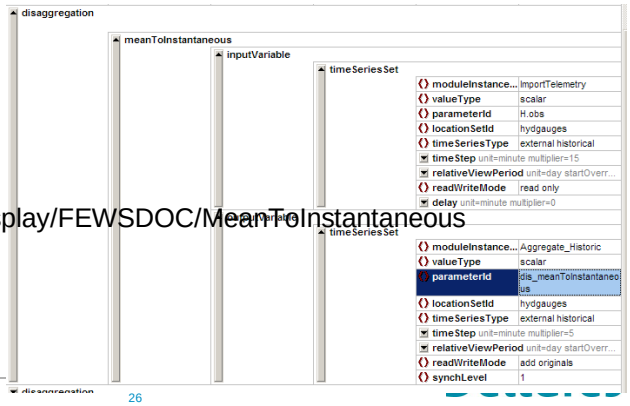
Deltares

Instantaneous



meanToInstantaneous

- This transformation takes a mean input time series as input and transforms it to an instantaneous time series.
- The transformation will make a best estimate of the original instantaneous time series.
- Takes into account:
  - Rise/Fall
  - Peak/Valley
- Details at: http://public.deltares.nl/display/FEWSDOC/MeanToInstantaneous (to be updated… ;-)



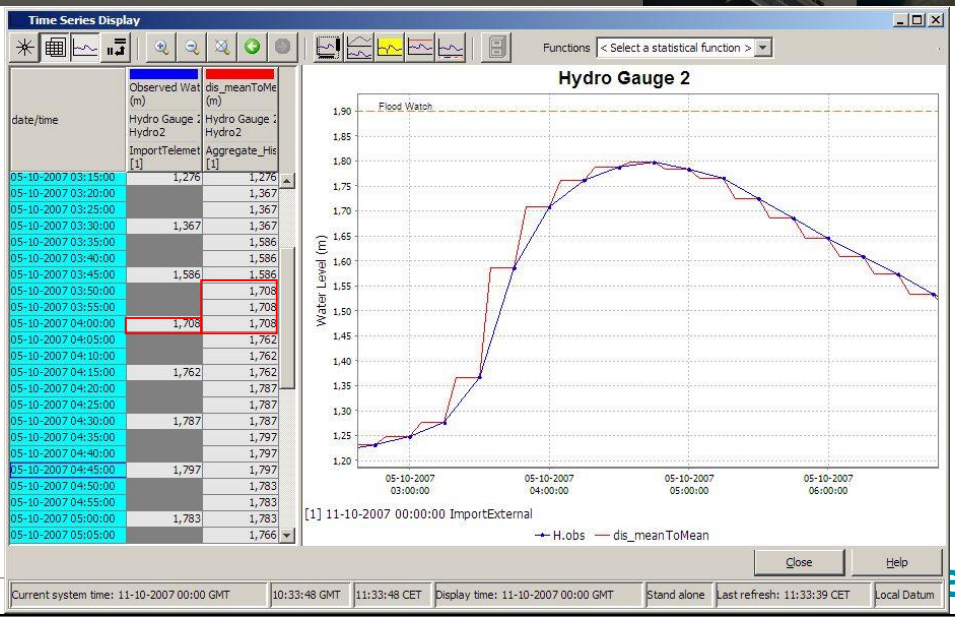
meanToMean

- This transformation performs a disaggregation from a mean time series to a mean time series.
- Each output time series value within a given data time interval of the input time series is equal to the input time series value for that interval.
- The table shows a simple example of the procedure.

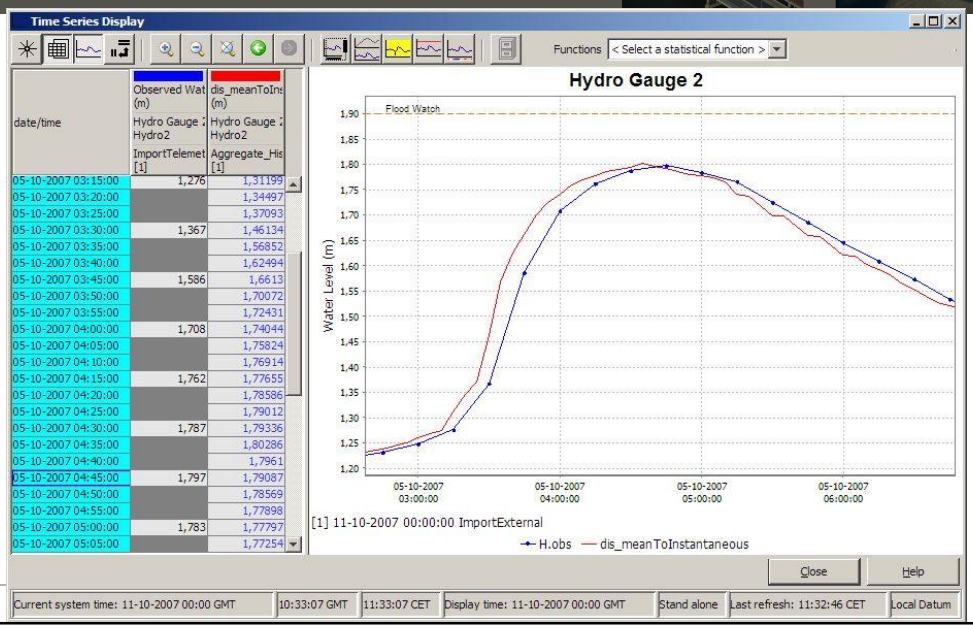
Time	Input	Output
12:00	x	1
00:00	1	1
12:00	x	2
00:00	2	2

Deltares

meanToMean



meanToInstantaneous

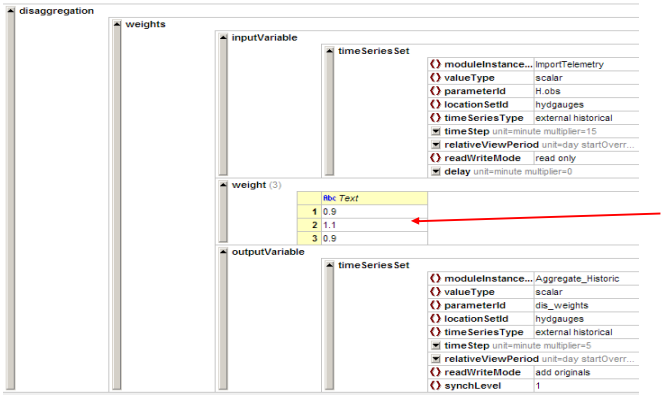


date/time	Observed Wat (m) Hydro Gauge 2 Hydro2 ImportTelemet [1]	dis_meanToIns (m) Hydro Gauge 2 Hydro2 Aggregate_His [1]
05-10-2007 03:15:00	1,276	1,31199
05-10-2007 03:20:00		1,34497
05-10-2007 03:25:00		1,37093
05-10-2007 03:30:00	1,367	1,46134
05-10-2007 03:35:00		1,56852
05-10-2007 03:40:00		1,62494
05-10-2007 03:45:00	1,586	1,6613
05-10-2007 03:50:00		1,70072
05-10-2007 03:55:00		1,72431
05-10-2007 04:00:00	1,708	1,74044
05-10-2007 04:05:00		1,75824
05-10-2007 04:10:00		1,76914
05-10-2007 04:15:00	1,762	1,77655
05-10-2007 04:20:00		1,78586
05-10-2007 04:25:00		1,79012
05-10-2007 04:30:00	1,787	1,79336
05-10-2007 04:35:00		1,80286
05-10-2007 04:40:00		1,7961
05-10-2007 04:45:00	1,797	1,79087
05-10-2007 04:50:00		1,78569
05-10-2007 04:55:00		1,77898
05-10-2007 05:00:00	1,783	1,77797
05-10-2007 05:05:00		1,77254

weights

Dis-aggregate using **weights** function. Multiplies input points with weight for each output point

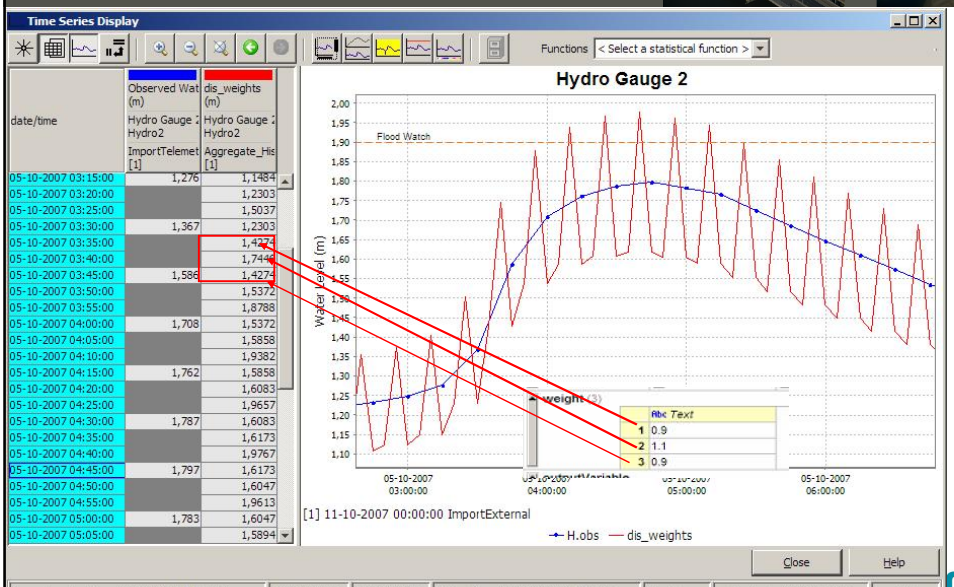
- For example used in dis-aggregating daily evaporation series



Nr of weights ***must*** match nr of new timesteps

Deltares

weights



Aggregation – Exercise 18

1. Aggregate 15 minute rainfall data to hourly series
2. Aggregate 15 minute waterlevel data to hourly series

3. Try to aggregate to daily series

Performance indicators

Deltares

Performance assessment

Performance assessment

- measures the quality/reliability of model simulations and model forecasts

Domains

- Model calibration and validation
- Forecast verification

In FEWS

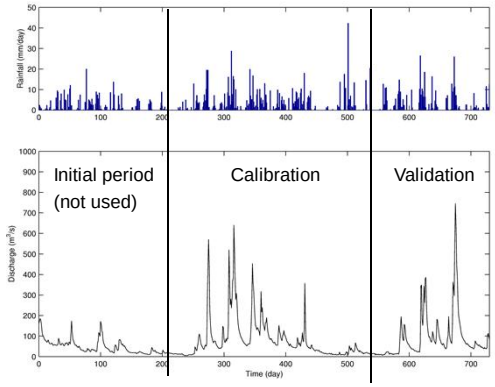
- Model calibration - performance assessment of models over previous time period (eg 24 hrs)
- Forecast verification – Lead time accuracy (avg over all forecasts) & skill scores

Deltares

Model calibration and validation

Objective: Establish a model that adequately simulates behaviour

- Calibration: Propose model structure & parameterisation
- Validation: Assess if proposed model/structure acceptable



Forecast verification

Goal: verify our forecast and uncertainty information.

Help us answer questions such as

- How good/reliable are the forecasts?
- How skillful are the forecasts?
- What are the strengths and weaknesses in the forecasts?
- What are the sources of uncertainty in the forecasts?

- How is new science improving the forecasts?
- What should be done to improve the forecasts?
- How well do forecasts meet targets that have been set?

Forecast skill

Forecast skill refers to the relative accuracy of a set of forecasts, with respect to some set of standard control, or reference, forecasts

Wilks, 2005

Examples of reference forecasts

- Climatological average
- Climatological variability
- Peristence

Deltares

Forecast verification is conditional

verification can be done in two ways

Conditional on the observations

- Answers the question: given an observation of an event of given magnitude, what is the probability that the forecast will be able to predict this

Conditional on the forecasts

- Answers the question: Given a forecast of an event of given magnitude, what is the probability that the observed value will indeed attain that magnitude

Deltares

Prognostic/Real-time vs Diagnostic

Diagnostic verification

- Verification of forecasts made off-line - Controlled environment
- Establish reliability of models etc.
- Isolate parts of chain
  - e.g. reliability of model forecasts using “perfect” rainfall

Prognostic / Real-Time verification

- Verification within the forecast process
- Statistics reflect all uncertainties (data, model, operational)

Deltares

Objective Functions & performance metrics

Objective functions – measurable criteria for attaining objective

Do not be confused!

- An objective function is not an *objective* view of how good the model is
- Selection of an objective function should reflect model purpose
  - > Selection of objective function is *subjective*

Objective function: statistic expressing comparison between modelled time series and observed time series
(hard measure)

Soft measures also possible – i.e. if model does not show this behaviour it is not acceptable

Deltares

Performance indicators – statistics!

Categories of performance indicators

Continuous simulation (calibration)

- Metrics: Bias, MAE, RMSE, N-S, R^2

Deterministic forecasts of continuous predictands

- Metrics: MAE, RMSE, R^2, *Scatter plots*, *Conditional quantile plots*

Deterministic forecasts of discrete predictands

- Metrics: Skill Scores: POD, FAR, CSI, *Bias*, *POFD, etc*

Probabilistic forecasts

- Brier (skill) scores, Ranked Probability Score, Reliability diagram, Talagrand plots, ROC etc.

Deltares

Objective functions

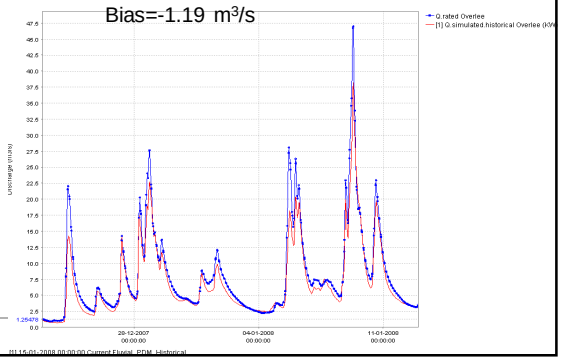
Bias (-Inf to +Inf)

$$BIAS = \frac{1}{N}\sum_{i}^{N} \hat{x}_i - x_i$$

Expresses average error

Answers question: Is my prediction on average too high or too low?

Bias=-1.19 m³/s



But:

- errors compensate
- no measure of error magnitude
- does not measure correspondence

Usage:

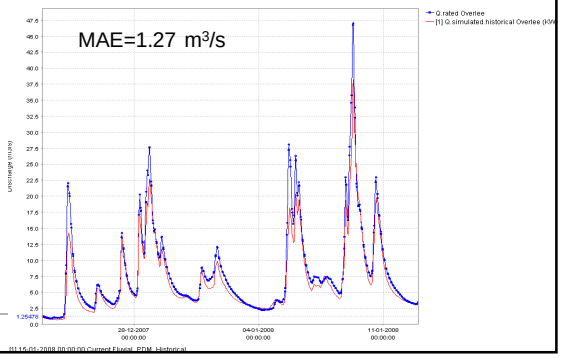
Water Resources

Objective functions

Mean Absolute Error (0 to +Inf)

$$MAE = \frac{1}{N}\sum_{i}^{N} | \hat{x}_i - x_i |$$

Answers question: What is the average magnitude of the errror?



But:

- does not measure error direction

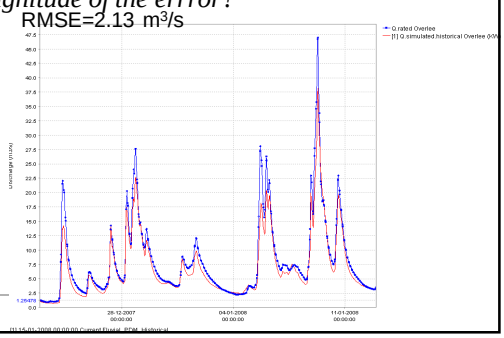
Objective functions

Root Mean Squared Error (0 to +Inf)

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\hat{x}_i - x_i\right)^2}$$

Measures "average" error, weighted according to the square of the error

Answers question: What is the average magnitude of the errror?



But:

- does not measure error direction
- greater emphasis on large errors

Usage:

Common for flood models

Objective functions

Correlation Coefficient (-1 to +1)

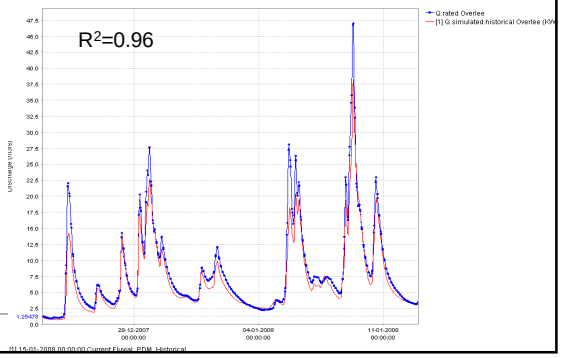
$$r = \frac{n\sum x\hat{x} - (\sum x)(\sum \hat{x})}{\sqrt{n(\sum x^2) - (\sum x)^2}\sqrt{n(\sum \hat{x}^2) - (\sum \hat{x})^2}} \qquad R^2 = r^2$$

Good measure of linear association or phase error

Answers question: How well did the simulated values correspond to the observed values?

But:

- does not consider bias
- sensitive to outliers



Objective functions

Nash-Sutcliffe (-Inf to +1)

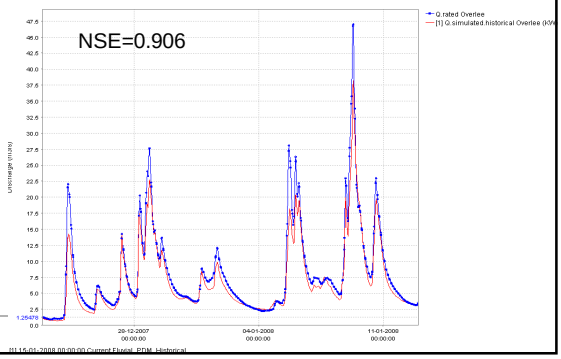
$$NS = 1 - \frac{\sum_i^N (\hat{x}_i - x_i)^2}{\sum_i^N (x_i - \overline{x})^2}$$

Measure very common in hydrological modelling

Answers question: What is the variance of the errors – scaled to the variance of the observations?

But:

- sensitive to bias
- unreliable for event forecasts



Objective functions

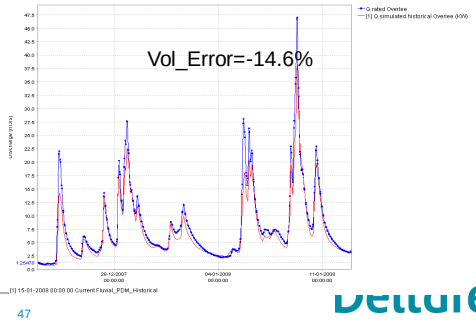
Volume error (%) (-Inf to +Inf)

$$PERC\_VOLUME = 100 \times \frac{\sum_{i=2}^{N} (\hat{x}_i + \hat{x}_{i-1})}{\sum_{i=2}^{N} (x_i + x_{i-1})}$$

Answers question: What is the difference in volume between simulation and observed?

Actually same as bias

Water resources

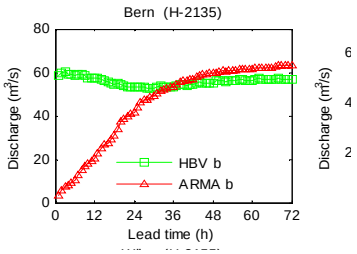


Deltares

Deterministic forecasts of continuous predictands

Lead time accuracy

- Assess accuracy at given lead time



Common statistics used

- Bias
- RMSE
- R^2

Answers the question: *What is the average (magnitude of the) errror at the selected lead time?*

Deltares

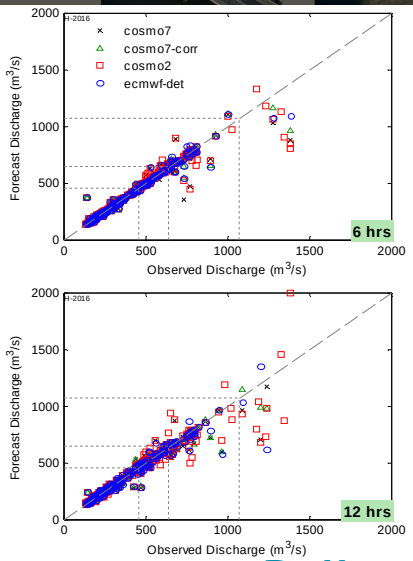
Deterministic forecasts of continuous predictands

Scatter plots

- Plot forecast values at selected lead time against observed
- Ideal forecast is on diagonal

Answers the question: How well did the forecast values correspond to the observed values?

- Easy to interpret visually
- Reveals relationship between magnitude of forecast values and errors



Deltares

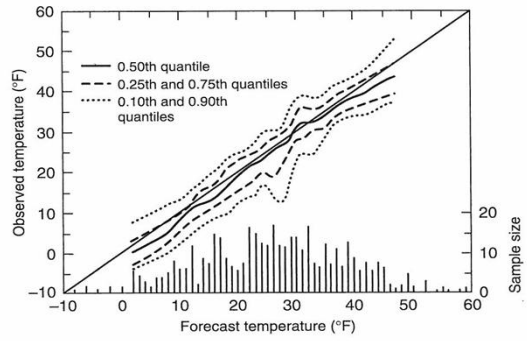
XXXXX Deterministic forecasts of continuous predictands

Conditional quantile plot

- plot distribution of observed values for each forecast value

Answers the question: Given a forecast value, what is the expected value of the observed

- Expression of expected bias in forecast
- Potentially very useful to forecaster



Deltares

Deterministic forecasts of discrete predictands

2 x 2 contingency table

	Threshold Observed	
Threshold Forecast	**Yes**	**No**
Yes	a	b
No	c	d

Sample size = a+b+c+d

Marginal totals

- Number of obs events = a+c
- Number of obs non events = b+d
- Number of forecast events = a+b
- Number of forecast non events = c+d

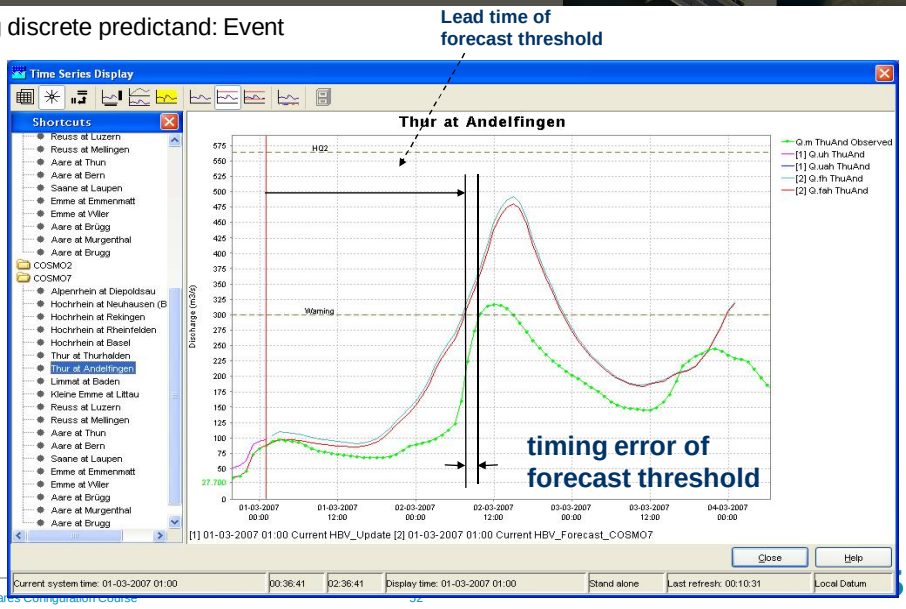
Marginal distributions – divide totals by sample size

Reference “climatology” (a+c)/n

Deltares

Deterministic forecasts of discrete predictands

Deriving discrete predictand: Event



Deterministic forecasts of discrete predictands

Percentage correct (0 to 1)

$$PC = \frac{a+d}{n}$$

	Threshold Observed	
Threshold Forecast	**Yes**	**No**
Yes	a	b
No	c	d

Answers the question: Overall, what fraction of the forecasts were correct?

Characteristics: Simple, intuitive. Can be misleading since it is heavily influenced by the most common category, usually "no event" in the case of rare weather

Deltares

Deterministic forecasts of discrete predictands

Bias (0 to +Inf)

$$Bias = \frac{a+b}{a+c}$$

	Threshold Observed	
Threshold Forecast	**Yes**	**No**
Yes	a	b
No	c	d

Answers the question: How did the forecast frequency of "yes" events compare to the observed frequency of "yes" events?

Characteristics: Measures the ratio of the frequency of forecast events to the frequency of observed events. Indicates whether the forecast system has a tendency to underforecast (BIAS<1) or overforecast (BIAS>1) events. Does not measure how well the forecast corresponds to the observations, only measures relative frequencies.

Deltares

Deterministic forecasts of discrete predictands

Probability of Detection (0 to 1)

$$POD = \frac{a}{a+c}$$

	Threshold Observed	
Threshold Forecast	**Yes**	**No**
Yes	a	b
No	c	d

Answers the question: What fraction of the observed "yes" events were correctly forecast?

Characteristics: Sensitive to hits, but ignores false alarms. Very sensitive to the climatological frequency of the event. Good for rare events. Can be artificially improved by issuing more "yes" forecasts to increase the number of hits. Should be used in conjunction with the false alarm ratio. POD is also an important component of the Relative Operating Characteristic (ROC) used widely for probabilistic forecasts.

Deltares

Deterministic forecasts of discrete predictands

False Alarm Ratio

$$FAR = \frac{b}{a+b}$$

	Threshold Observed	
Threshold Forecast	**Yes**	**No**
Yes	a	b
No	c	d

Answers the question: What fraction of the predicted "yes" events actually did not occur (i.e., were false alarms)?

Characteristics: Sensitive to false alarms, but ignores misses. Very sensitive to the climatological frequency of the event. Should be used in conjunction with the probability of detection

Deltares

Deterministic forecasts of discrete predictands

Critical Success Index / Threat Score (0 to 1)

$$CSI = \frac{a}{a+b+c}$$

Threshold Forecast	Threshold Observed: Yes	Threshold Observed: No
Yes	a	b
No	c	d

Answers the question: How well did the forecast "yes" events correspond to the observed "yes" events?

Characteristics: Measures the fraction of observed and/or forecast events that were correctly predicted. It can be thought of as the accuracy when correct negatives have been removed from consideration, that is, CSI is only concerned with forecasts that count. Sensitive to hits, penalizes both misses and false alarms. Depends on climatological frequency of events (poorer scores for rarer events) since some hits can occur purely due to random chance.

Deltares

Deterministic forecasts of discrete predictands

Probability of False Detection (1 to 0)

$$POFD = \frac{b}{b+d}$$

Threshold Forecast	Threshold Observed: Yes	Threshold Observed: No
Yes	a	b
No	c	d

Answers the question: What fraction of the observed "no" events were incorrectly forecast as "yes"?

Characteristics: Sensitive to false alarms, but ignores misses. Can be artificially improved by issuing fewer "yes" forecasts to reduce the number of false alarms. Not often reported for deterministic forecasts, but is an important component of the Relative Operating Characteristic (ROC) used widely for probabilistic forecasts.

Deltares

Deterministic forecasts of discrete predictands

Some remarks:

- Scores presented are marginal – joint distribution of forecasts and observations is what we want!
  - Which means – a single score is insufficient to provide a complete answer
  - Consider 3 attributes (e.g POD, POFD, Climatological frequency)
- Are verification results trustworthy (sampling uncertainty)?
  - particularly for higher thresholds
- Many methods pioneered in meteorology – applicability to hydrology not always sure due to different nature of predictand.

Deltares

Performance assessment with FEWS

Prognostic – Real Time

- Configure to run as a workflow – e.g. daily
- Build statistics through time
  - Improvements visible (annual trend)
  - Performance versus magnitude of events
- Measure of service – including all uncertainties

Deltares

Performance assessment with NFFS/FEWS

Diagnostic – Offline

- Use Batch mode facility
- Can run through multiple years of forecasts
- Build database from scratch
- Configure changes to data processing – e.g. use perfect rainfall forecast
- Configure performance indicators (RVP's) differently to suit data used

Steps:

- Start from empty database
- Load data
- Run batch of historic events at e.g. daily step – create history
- Run batch of forecasts (full period, above threshold)
- Assess results in FEWS, export for more complete assessment

Deltares

References

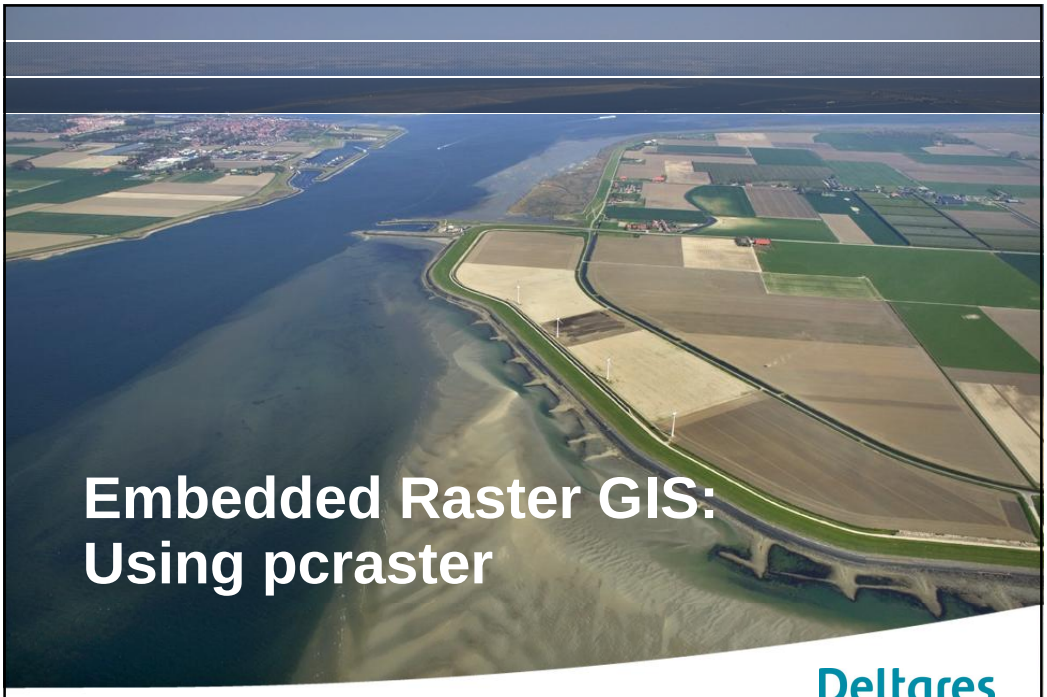
Forecast Verification - Issues, Methods and FAQ, WMO
http://www.bom.gov.au/bmrc/wefor/staff/eee/verif/verif_web_page.html

Statistical methods in the Atmospheric Sciences, David S. Wilks
International Geophysics Series, 2nd Edition, 2005

Fews Documentation:
http://public.deltares.nl/display/FEWSDOC/10+Performance+Indicator+Module
http://public.deltares.nl/display/FEWSDOC/07+Skill+Score+Display

Example (in Data directory):
Performance_Example 1.00 default.xml

Deltares



Embedded Raster GIS: Using pcraster

Deltares

pcraster and Delft-Fews

From the pcraster web-site (www.pcraster.nl)

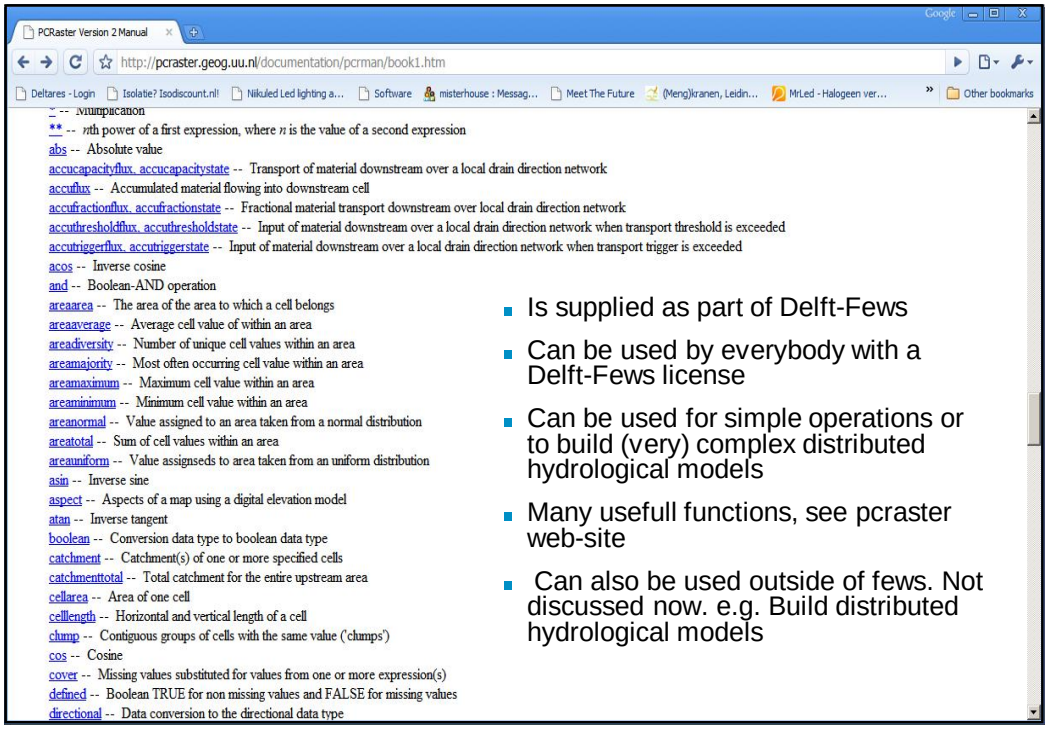
The PCRaster Environmental Modelling language is a computer language for construction of iterative spatio-temporal environmental models. It runs in the PCRaster interactive raster GIS environment that supports immediate pre- or post-modelling visualisation of spatio-temporal data.

The PCRaster Environmental Modelling language is a high level computer language: it uses spatio-temporal operators with intrinsic functionality especially meant for construction of spatio-temporal models.

Key concepts:

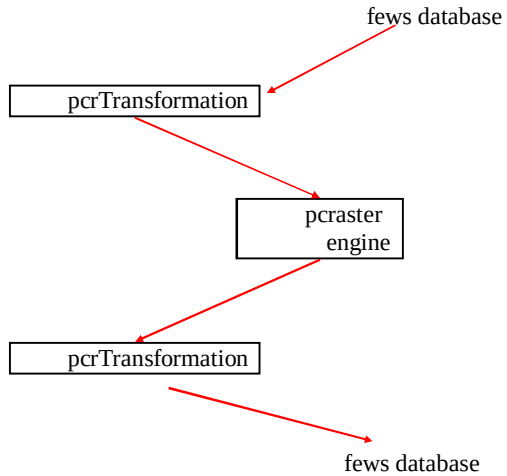
- Script language for gridded data
- many hydrological functions (e.g. kinematic wave, catchment deliniation etc)
- Integrated into Delft-Fews using in-memory XML link (external link via general adapter also possible)

Deltares



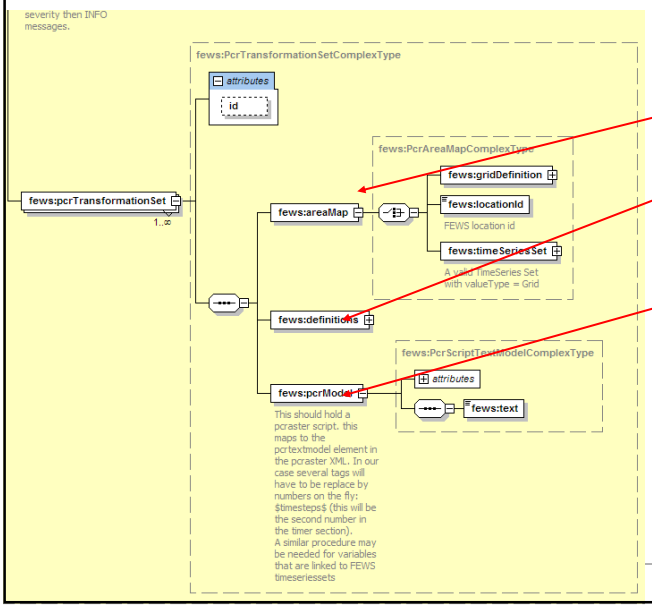
Link with FEWS

- In memory XML based interface
- Fast and open API, also linked to openDA (in progress) and openMI



Deltares

The pcrtransformation module



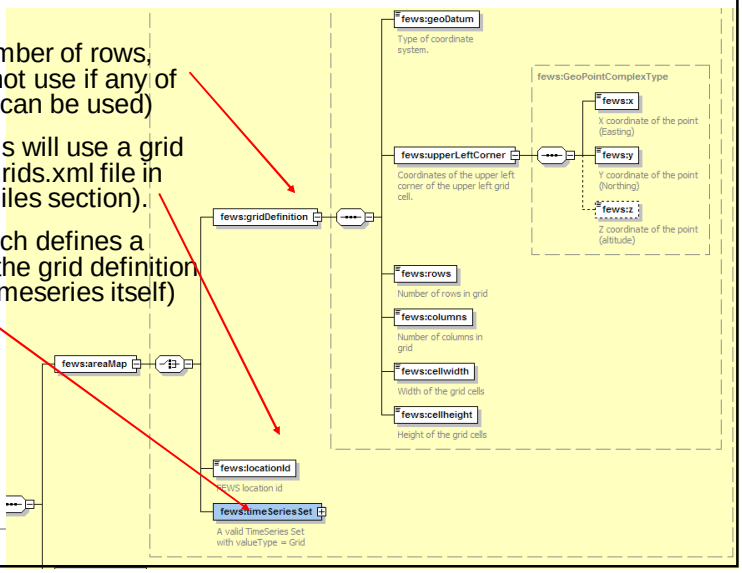
- Define the grid (**areaMap**)
- Define in and output (**definitions**: what to send to script and what to recieve from script; timeseriessets)
- Define the script (**pcrModel**)
- Multiple transformation/models in one file

Deltares

The pcrtransformation module

The area map can be defined using three methods:

- Grid Definition (number of rows, columns etc.) (do not use if any of the other methods can be used)
- Grid location Id (this will use a grid definition the the Grids.xml file in the RegionConfigFiles section).
- TimeSeriesSet which defines a Grid TimeSeries. (the grid definition is taken from the timeseries itself)



Example

- The simple code below is a regression based groundwater model

Lister - [D:\WaterBudget_Forecast 1.00 default.xml]

File Edit Options Help 53 %

```
  <pcrModel id="GWMODEL">
   <text><![CDATA[
#! --unittrue --degrees
#  Peat Water Budget Model
# Inputs from Delft-Fews into this script
# P -> daily precip
# GWOLD -> Groundwater Level from previous run (used as initial condition with 1 day delay)
#
initial
 GWinit = cover(GWOLD,0.0) * scalar(PEAT);
    Pinit = spatial(0.0);
 ETinit = spatial(3.8);
 Sf = spatial(0.26);

dynamic
    PP = cover(P,0.0); # Fill in with zeros if missing
    PP = if(PP >= 0 then PP else spatial(0.0)); # remove negative P (due to bias correction)
 GW = min(0.0,if (GWinit <= 0 then GWinit + (Pinit - ETinit)/Sf/1000.0 else 0.0));
 ET = if(GWinit > -1 then 3.8 else (1.97 * GWinit) + 5.75);

 ETinit = ET;
 Pinit = PP;
 GWinit = GW;
]]></text>
   </pcrModel>
  </pcrTransformationSet>
```

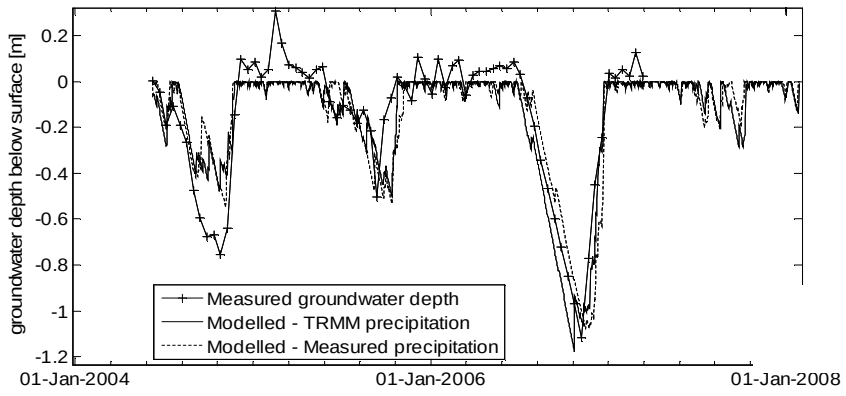
Initial section (optional), only executed at start

Dynamic section, executed each timestep

Deltares

Example

- Result



Deltares

Rainfall polygons

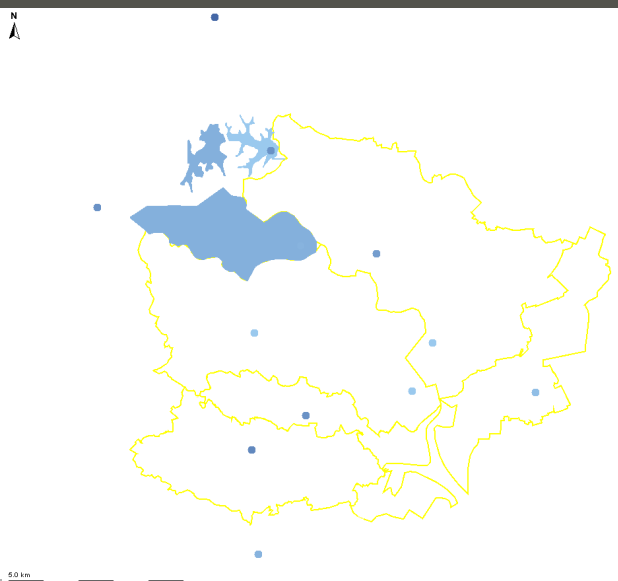
Input from FEWS
FEWS puts value of the scalar at the proper location in the grid

```
# Simple Theisen polygons to get spatial average precipitation on a grid
# Creat unique Id's for input stations
Unq = uniqueid(boolean(P));
# Now generate polygons and fill those
GaugeArea = spreadzone(ordinal(cover(Unq,0)),0,1);
MeasMap = areaaverage(P,GaugeArea);
```

– Output back to FEWS

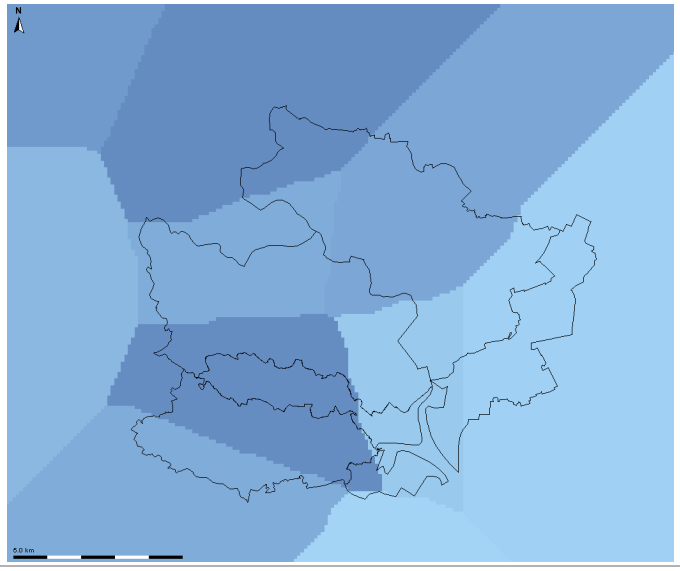
Deltares

Rainfall polygons - Input



Deltares

Rainfall polygons - output



Deltares

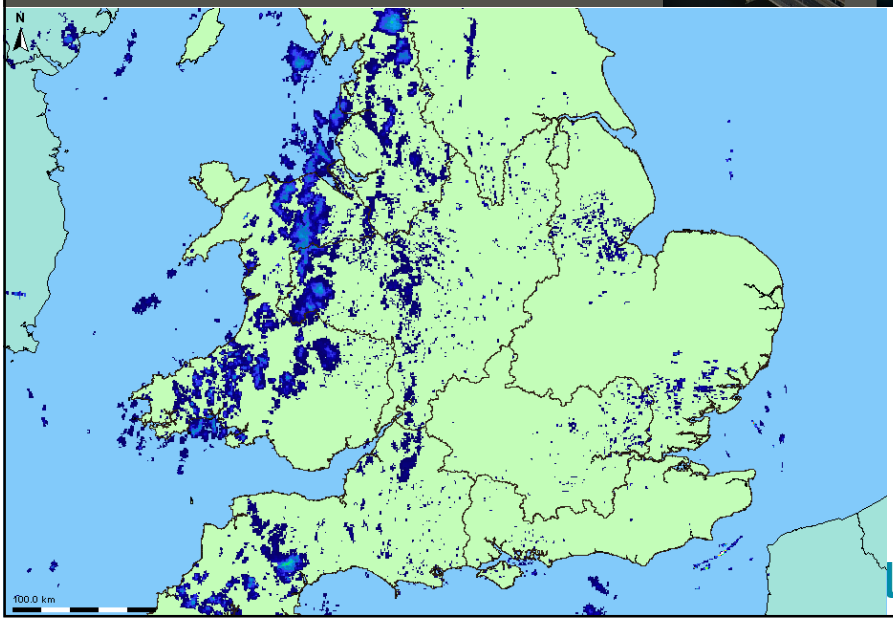
Filter radar data

```
#! --unitcell

dynamic
  RADARunit = if(Radar > 0.0 then 1.0);
   RF=windowtotal(RADARunit,2);
   RFL = windowtotal(RADARunit,6);
   RADARFILT = if(RF > 2 or RFL > 14 , Radar);
```

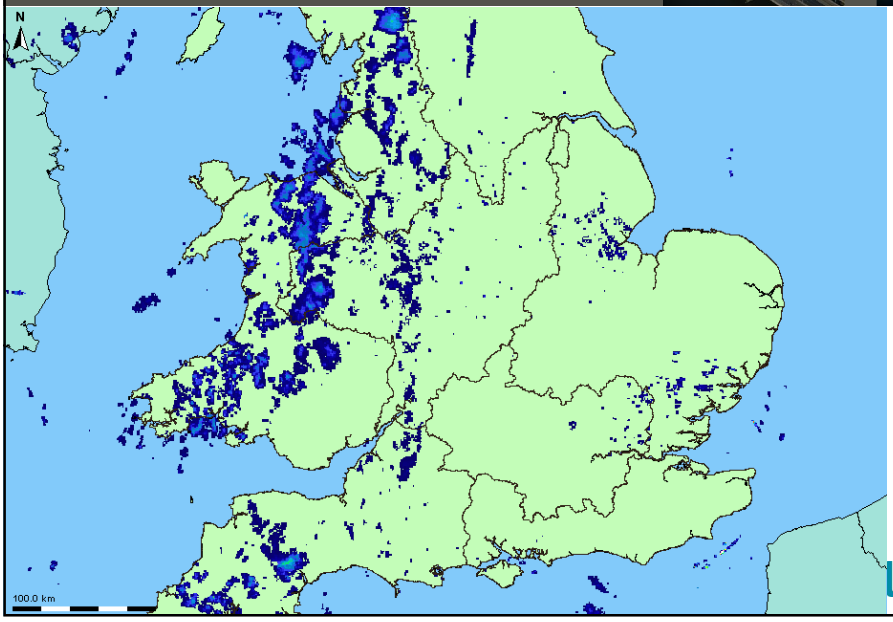

Deltares

Filter radar data



ltares

Filter radar data



ltares

How to use/Configure this module

- Module needs to be know in Delft-Fews -> Add to moduledescriptors

What	*nameofinstrance*.xml
Description	Configuration of the pcraster transformation module
schema location	http://fews.wldelft.nl/schemas/version1.0/pcrTransformationSets.xsd
Entry in ModuleDescriptors	<moduleDescriptor id="PcrTransformation"> <description>PCr Transformation Component</description> <className>nl.wldelft.fews.system.plugin.transformation.PcrTransformationController</className> </moduleDescriptor>

- Make your moduleinstance: documentation at:

http://public.deltares.nl/display/FEWSDOC/16+Pcraster+Transformation+(pcrTransformation)

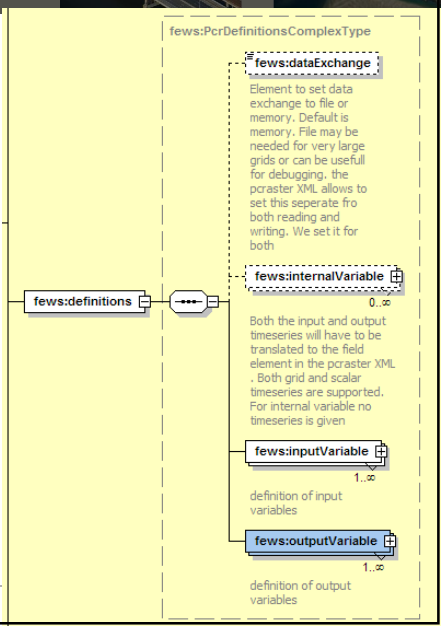
- Add to moduleInstancedescriptors (and worklow)
- Run!

Deltares

The pcrtransformation module

- Data Exchange Type- memory or file (use memory!)
- Internal PCRaster model variable, with a unique Id and data type. The data type (which are exactly similar to the data types used in PCRaster) can be (usually not needed as variable may be define in the script on the fly)
- Input PCRaster model variable. Use to get data from Delft-Fews and pass it on to the pcraster transformation module:

- Variable id (should be matching exactly as defined in the PCRaster text model),
- Data type (similar to that used for internal variables),
- Scalar type data to be passed to PCRaster, if different from the normal data value.



The pcrtransformation module

Solar radiation example

open pcrfews.exe

Deltares

Day 4 afternoon

- Client-server configuration issues
- Sobek and the Sobek Adapter
- Extra topics time permitting

Deltares

Client-Server Configuration Issues

Deltares

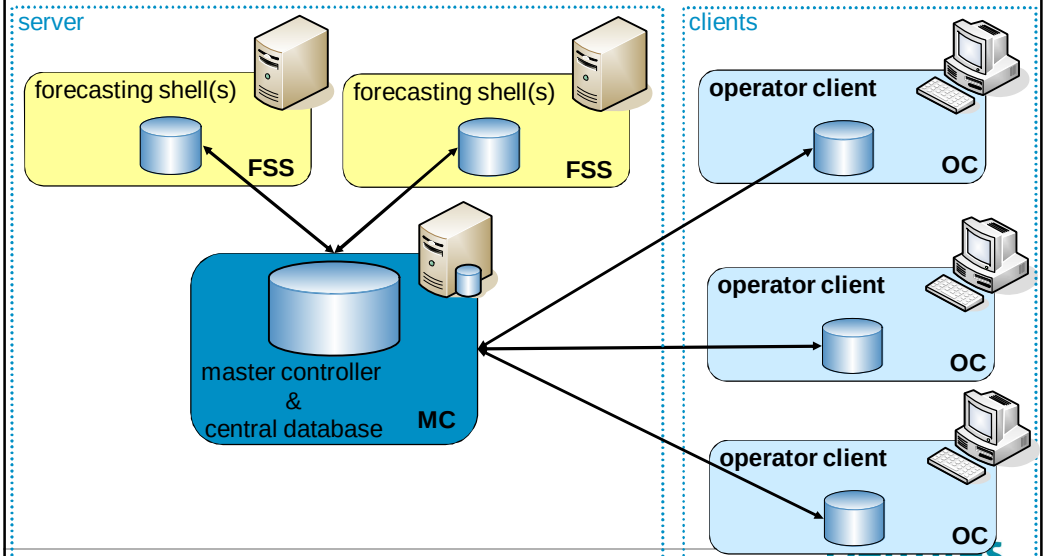
TimeSeries in a Client-Server system

A FEWS Client Server System uses the same configuration as a Stand Alone system

- When updating the configuration of a FEWS application, always use the Stand Alone system
- It is good to know how the Client-Server FEWS system works
- The configation has some special features with respect to:
  - Synchronisation
  - Expiry time
  - Precision used to store the data

Deltares

Delft-FEWS system overview (client-server)



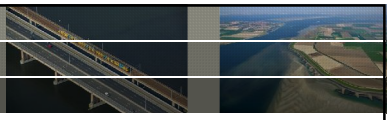
Synchronisation - FSS

Forecasting Shell(s)

- at startup (downloading)
  - build DB from scratch (new system, DB deleted)
- before a task (downloading)
  - latest config
  - latest data
- after a task (uploading)
  - created data

Deltares

Synchronisation - OC

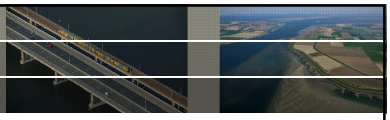


Forecasting Shell(s)

- at startup (downloading) / at login
  - build DB from scratch (new system, DB deleted)
  - configuration
  - data
- “on line”
  - at certain intervals (configured)
    - > data
    - > configuration
  - when a forecast is approved (by somebody else/system)

Deltares

Time Series Sets



Properties of time series:

- location Id – or – locationSetId
- parameter Id
- timestep
- relativeViewPeriod

- valueType (scalar, grid, profile, ..)
- timeSeriesType (external historical, external forecast, ..)
- moduleInstanceId (import, PDM, KW, ISIS, ..)
- readWriteMode

- expiryTime (when to delete from database)
- synchLevel

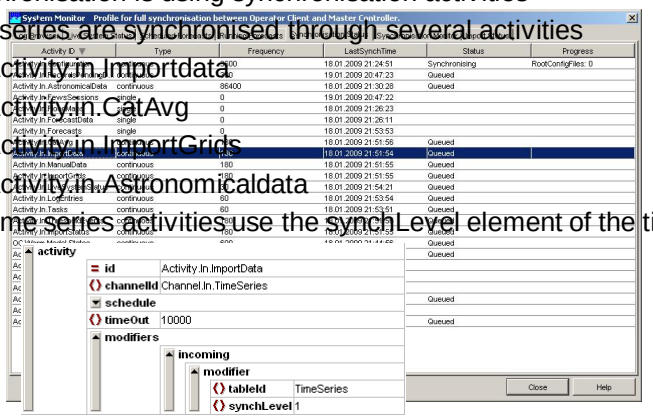
- simple transformations (delay, multiplier, divider, incrementer)
- ensembleId

Deltares

FEWS Synchronisation

The FEWS OC local datastore is retrieving data from the central database through synchronisation (using JMS)

- synchronisation is using synchronisation activities
- time series are synchronised through several activities
  - Activity.In.Importdata
  - Activity.In.CatAvg
  - Activity.In.ImportGrids
  - Activity.In.Astronomicaldata
- the time series activities use the synchLevel element of the timeSeriesSet



Deltares

Synchronisation

- Synchronisation Activities (mostly decribing for recognisition)
  - frequency
  - which 'Channel'
- SynchChannels
  - table
  - exceptions/modifiers
- SynchProfiles (organisation of Channels)
  - FSS profile
  - OC
    - > minimum
    - > full
    - > custom
  - CM

Deltares

Synchronisation Levels

synchLevels – selected by convention

0 (Default) All data from a forecast run
1 Scalar time series imported from telemetry
Data transformed using “fixed” relationship (Q-h, correlation)
2 All grid data from a forecast run (e.g. Flood Maps)
3 CatAvg data (forecasts, actuals & NWP)
4 Astronomical
5 Data edited on OC
6 Grid data imported from external forecast (synch. to OC)
7 Grid data imported from external forecast (FSS & MC only)
8 Performance indicators
9 Temporary time series (FSS only)

Deltares

Limits of FEWS Databases

Central database – ORACLE, PostGress, SQLServer
- No size limit

Forecasting Shell databases
- Microsoft ACCESS: 2 GB size limit
- Firebird: No size limit

Operator client databases
- Microsoft ACCESS: 2 GB size limit
- Firebird: No size limit

Of course: performance will reduce when database is larger

Deltares

Expiry Times

The total volume of data in the database can be reduced by :

- Configuring the expiry time
- Reducing the number of decimals for data values
- Mark time series as temporary data (not synchronised)

The maintenance task RollingBarrel deletes all expired data

- From the Central database (every 30 minutes)
- From the Forecasting Shells (every 24 hours)
- From the Operator Clients (every 30 minutes)

Deltares

Configure valueResolution of parameters

- By default a value is stored using 8 decimals
- Most observed data are imported with only 2-3 decimals
- The Parameters XML file allows configuration of **valueResolution**
- This can reduce the size of the data volume considerably

	= id	{} parameterType	{} unit	{} usesDatum	{} valueResolution	{} parameter
1	Discharge	instantaneous	m3/s			parameter (16)
2	Water Level	instantaneous	m	true		parameter (33)
3	Precipitation	accumulative	mm		0.01	parameter (19)
4	Temperature	instantaneous	oC		0.1	parameter (17)
5	Evaporation	accumulative	mm		0.1	parameter (7)
6	Radiation	instantaneous	W/m2		0.1	parameter (2)
7	Snow Rate	instantaneous	mm/hr		0.1	parameter (1)
8	Snow Depth	accumulative	mm		0.1	parameter (7)
9	Snow Weight	accumulative	g		0.1	parameter (1)
10	Snow Factor	instantaneous	%		0.1	parameter (3)
11	Wind Speed	instantaneous	m/s		0.1	parameter (8)
12	Wind Force	instantaneous	Bft		1	parameter (3)
13	Wind Direction	instantaneous	degrees		1	parameter (3)
14	Wave Height	instantaneous	m		0.1	parameter (7)

Deltares

Database size and synchronisation optimisation

Size of the database is mainly determined by:

- Size of the Simulated Forecast time series or External Forecast time series
- Number of forecast runs a day
- Length of time a forecast is held in the database (expiryTime)

Size of Simulated Forecast time series can be limited by:

- Change Expiry Time for less important time series or grids
- Not synchronising or storing all time series, only what you want to see on the OC
- Use valueResolution for parameter groups