



The Rhine Case

A compact, end-to-end example comparing multiple models for the Rhine by running one integrated verification pipeline. Learn the core *veriflow* workflow and how to interpret the resulting verification metrics.



Rijkswaterstaat
Ministerie van Infrastructuur
en Waterstaat

Deltares

VORTECH

SWIFT
SWFI
RESPONSE

Forecast verification with *veriflow* notebooks

Crash course + hands-on
exercise

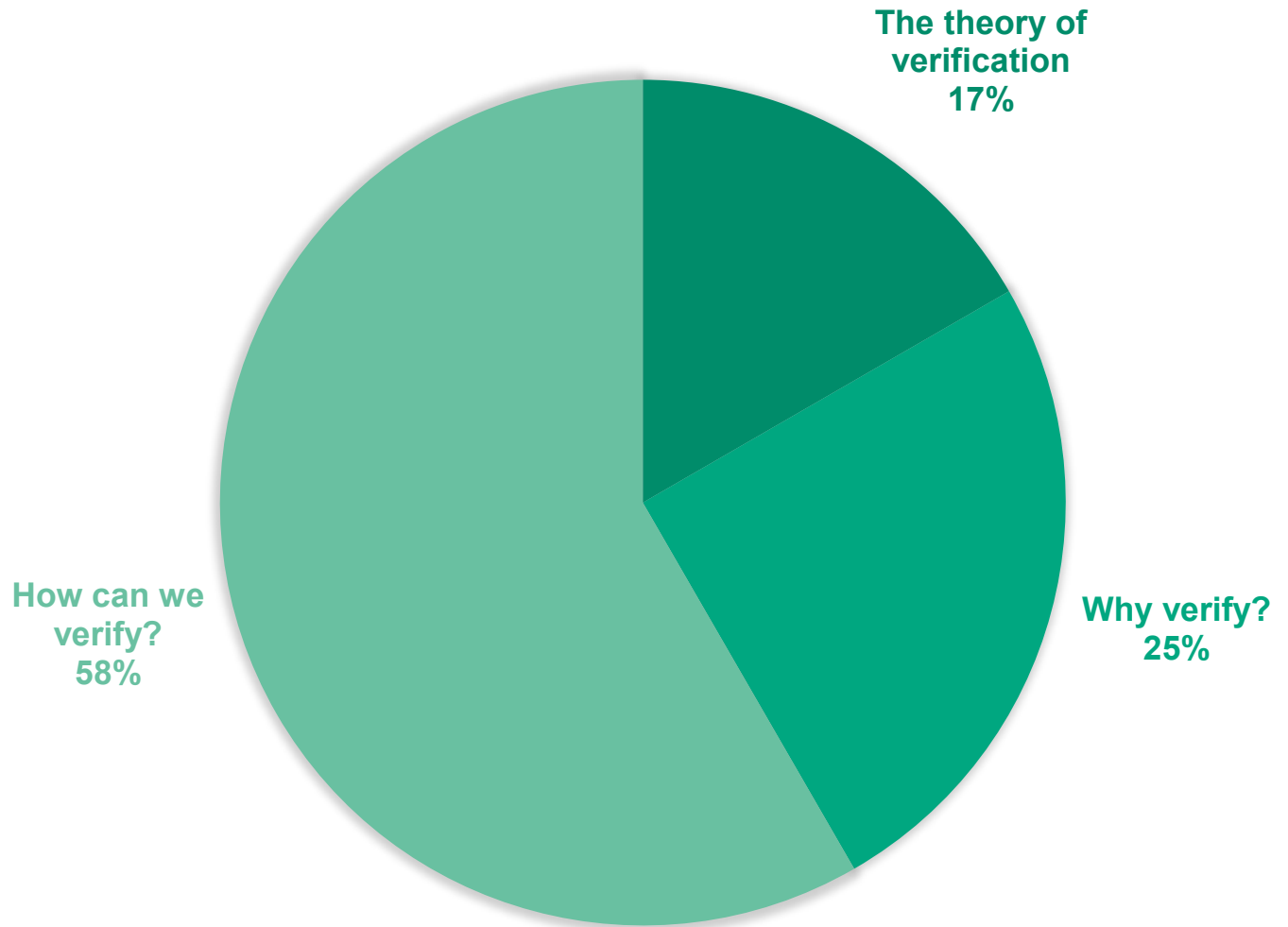
**Maarten Smoorenburg, Jurian Beunk, Cees
Voeselek, Aljen Uitbeierse, Dave Casson, Joost
Driebergen**

3 July 2026, Delft-FEWS Anwendertreffen @
LANUK Duisburg

Agenda

1. Two crash courses
 - Probabilistic forecasting
 - Forecast verification
2. Veriflow
3. Hands-on work
4. Discussions

The balance of this training



Focus on demo

You will each run your own verification pipelines

You will learn how to use Delft-FEWS for data access

You will each make your own visualisations

We will not: Spend a lot of time on in-depth theory of verification

Instead:

- We will cover some probabilistic scores
- We will share good references to theoretical background

By the end of this morning, you will be able to:

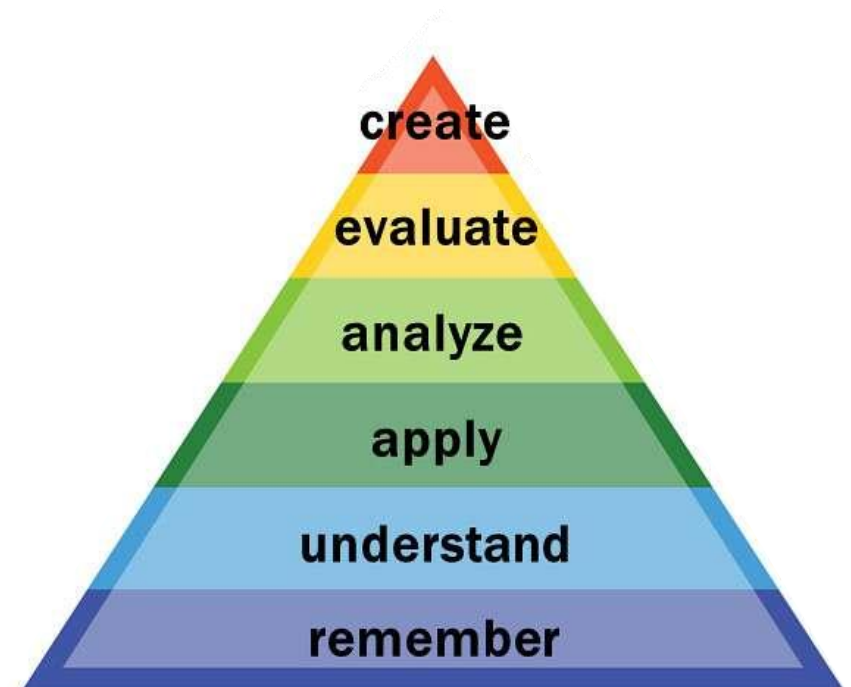
Understand the basics of verification and it's value

Understand how *veriflow* works and how it integrates with Delft-FEWS

Understand how *veriflow* can be applied in both research, model development and operations.

Run your own verification pipelines with *veriflow*

Make your own verification plots and visualisations





1 Crash Course: Probabilistic Forecasting

Why? What is it?

Learn more?

10

Nov

Probabilistic Forecasting - Short Course (DSD-INT 2026)

10 Nov 2026

09:00 - 16:30 (GMT+01:00)



Course (on-premises)

€ 629 (excluding 21% VAT)

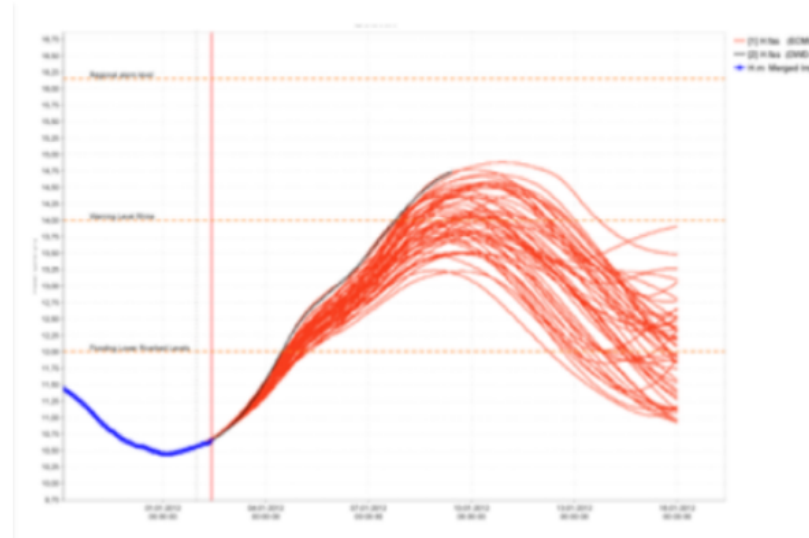
Room : Yangtze River Delta

Remaining places : 15



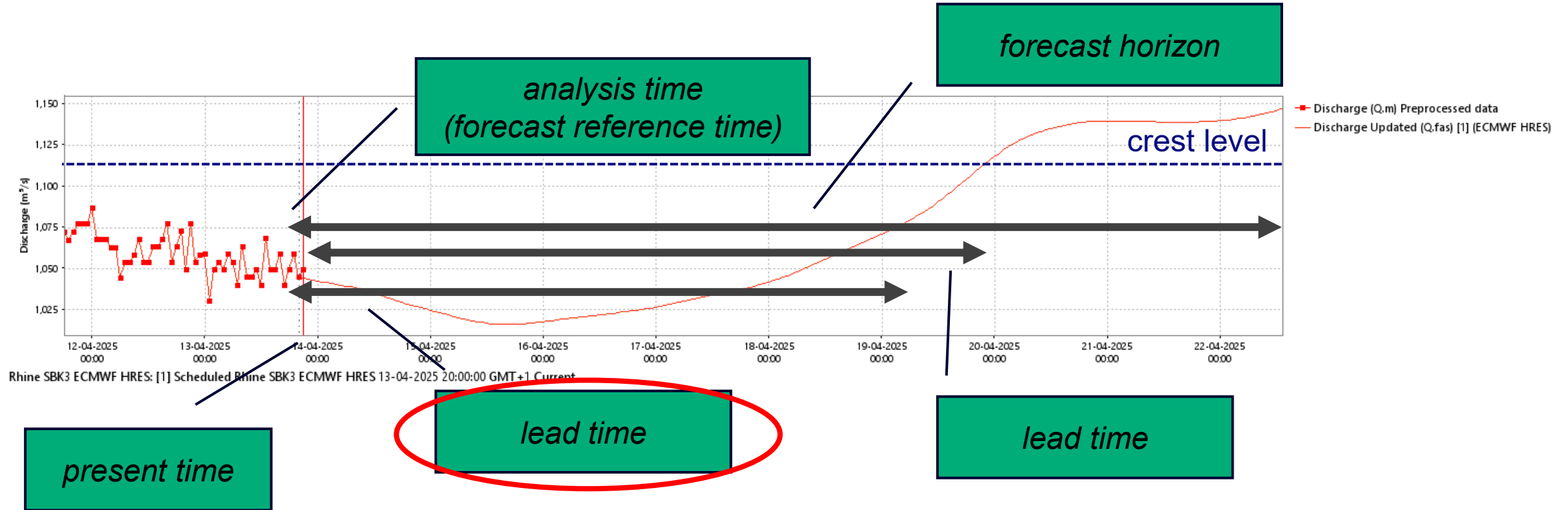
[Academy](#)

+ Add to my Calendar



[Probabilistic Forecasting - Short Course \(DSD-INT 2026\) - Academy](#)

Definitions



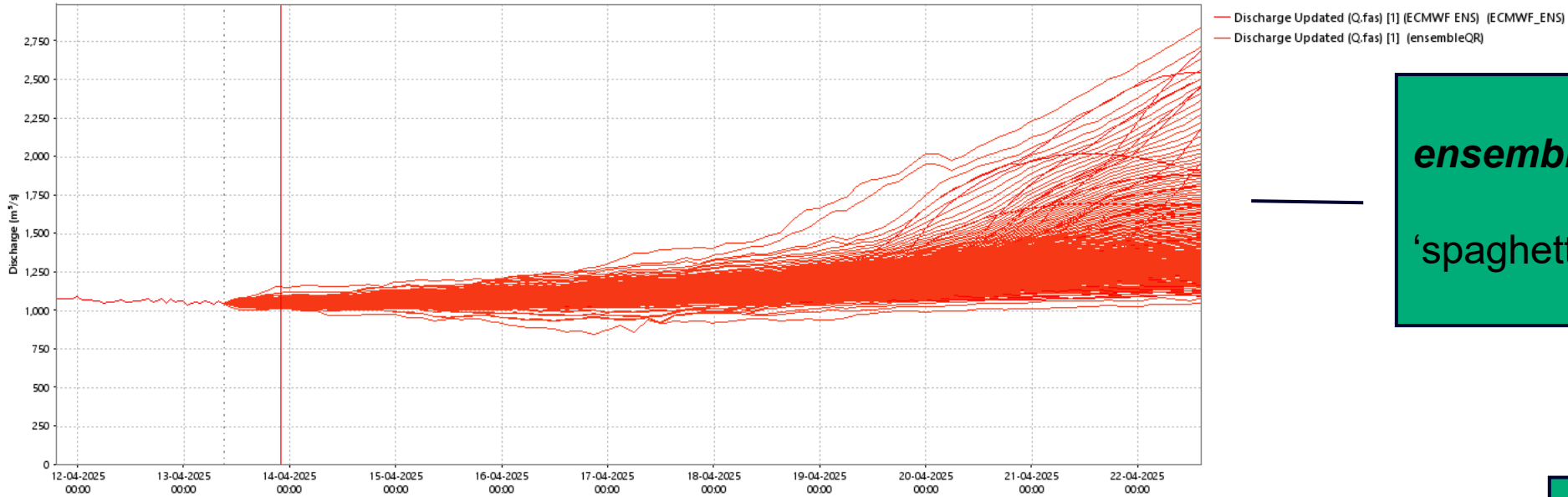
Other times:

Completion time: moment that forecast computation was finished

Issue time: time of publishing the (validated) forecast

Definitions

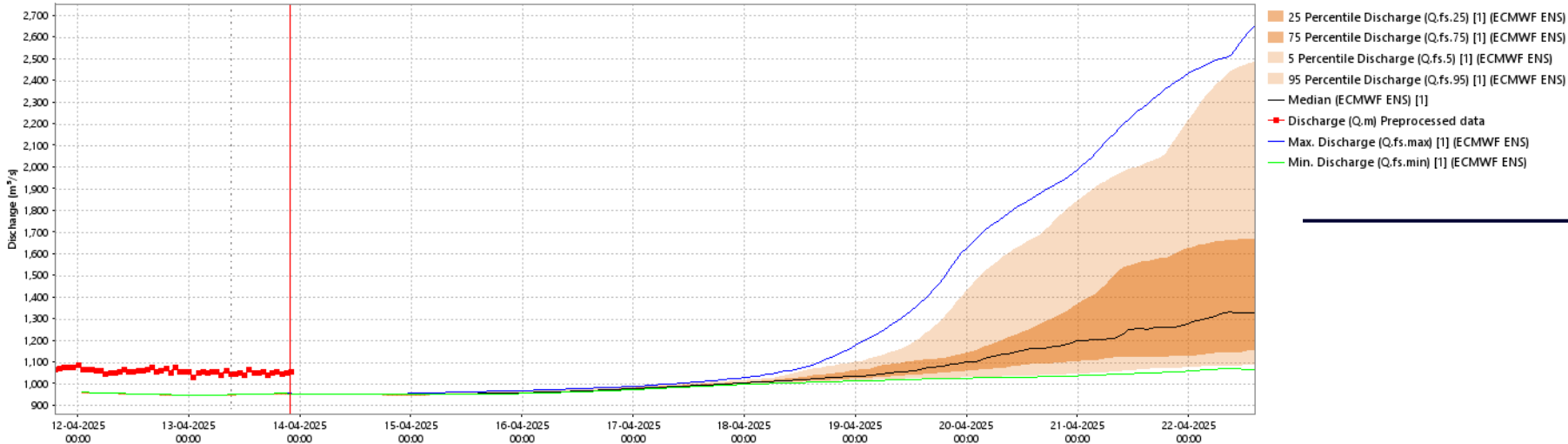
Lobith



ensemble forecast

'spaghetti plot'

Rhine SBK3 ECMWF ENS: [1] Scheduled Rhine SBK3 ECMWF ENS 13-04-2025 09:00:00 GMT+1 Current



probabilistic forecast

with:
bands / intervals
quantiles/percentiles (of exceedance)

Rhine SBK3 ECMWF ENS: [1] Scheduled Rhine SBK3 ECMWF ENS 13-04-2025 09:00:00 GMT+1 Current

Why use probabilistic forecasts?

Why use probabilistic forecasts?

1. Future is uncertain → show this explicitly
2. Enable risk-based decision-making → risk = *probability* x *consequences*
3. Extending forecast lead time
4. Allows for separation of responsibilities between forecaster and decision-maker



2 Crash Course: Forecast Verification

Why? What is it? With a focus on probabilistic forecasting

Verification is about measuring quality of models and forecasts



Different attributes of quality exist



The quality of a simulation or forecast is often measured with respect to a “true” or “observed” value

Streamflow or waterlevel measurements
Raingauges and temperature measurements



And can also be expressed relative to another simulation or forecast

Why is verification important for the operational domain?

Real time monitoring

- Continuous monitoring of models to detect deteriorating model performance. Relevant for re-calibration and re-training of models.

Improving the forecast

- Continuous decision support in real-time forecasting

Model improvement

- Advanced insights in forecast performance and quality
- Comparing model performance with different meteorological forcing

Reporting

- Reporting on model and forecast performance
- Management KPIs



What is a good forecast?

Murphy, 1993



Quality

the correspondence between the forecasts and the matching observations



Value

the incremental economic and/or other benefits realized by decision makers through the use of the forecasts

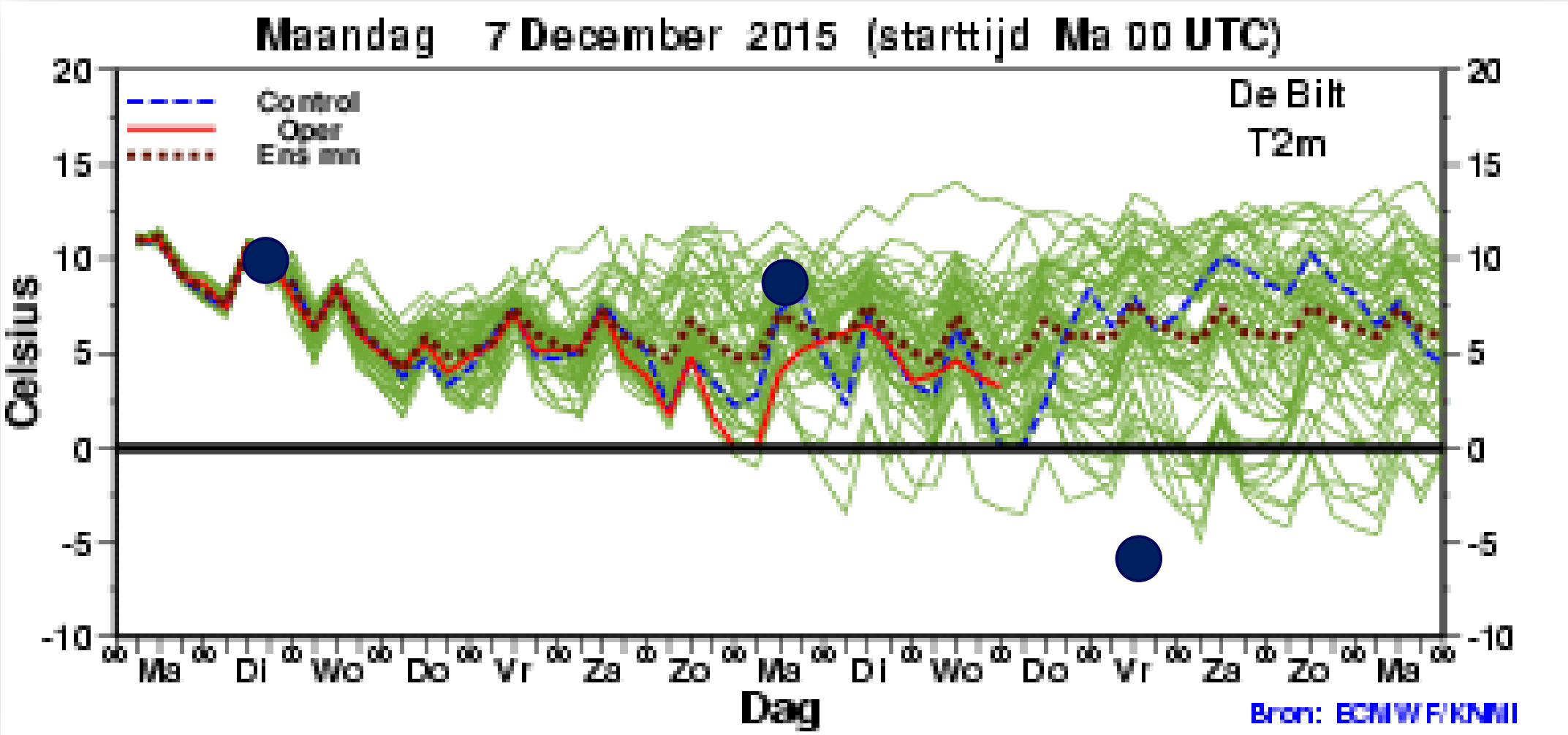


Consistency

the correspondence between forecasters' judgments and their forecasts

Murphy, Allan H. "What is a good forecast? An essay on the nature of goodness in weather forecasting." *Weather and forecasting* 8.2 (1993): 281-293.

How good or bad is this forecast?



What to expect from a probability forecast?

Accuracy

- Average correspondence between observed and predicted values

Reliability

- Correspondence of predicted probabilities with observed relative frequencies

Sharpness

- Tendency to produce 0% and 100% probability forecasts
- A sharp ensemble forecast has a small width

“Sharpness, given reliability”

Attributes of forecast quality

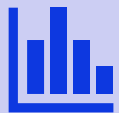
TABLE 2. Short definitions and relevant distributions for various aspects of forecast quality.

Aspect	Definition	Relevant distribution(s)
Bias	Correspondence between mean forecast and mean observation	$p(f)$ and $p(x)$
Association	Overall strength of linear relationship between individual pairs of forecasts and observations	$p(f, x)$
Accuracy	Average correspondence between individual pairs of forecasts and observations	$p(f, x)$
Skill	Accuracy of forecasts of interest relative to accuracy of forecasts produced by standard of reference	$p(f, x)$
Reliability	Correspondence between conditional mean observation and conditioning forecast, averaged over all forecasts	$p(x f)$ and $p(f)$
Resolution	Difference between conditional mean observation and unconditional mean observation, averaged over all forecasts	$p(x f)$ and $p(f)$
Sharpness	Variability of forecasts as described by distribution of forecasts	$p(f)$
Discrimination 1	Correspondence between conditional mean forecast and conditioning observation, averaged over all observations	$p(f x)$ and $p(x)$
Discrimination 2	Difference between conditional mean forecast and unconditional mean forecast, averaged over all observations	$p(f x)$ and $p(x)$
Uncertainty	Variability of observations as described by distribution of observations	$p(x)$

Murphy, A.H. 'What Is a Good Forecast? An Essay on the Nature of Goodness in Weather Forecasting'. *Weather and Forecasting* 8, no. 2 (1993): 281–293.

Today we use CRPS and the Rank Histogram

... but won't go into the theory behind it



Rank histogram

Captures reliability

[scores tutorial](#)



Continuous Ranked Probability
Score

Captures accuracy, reliability and
sharpness

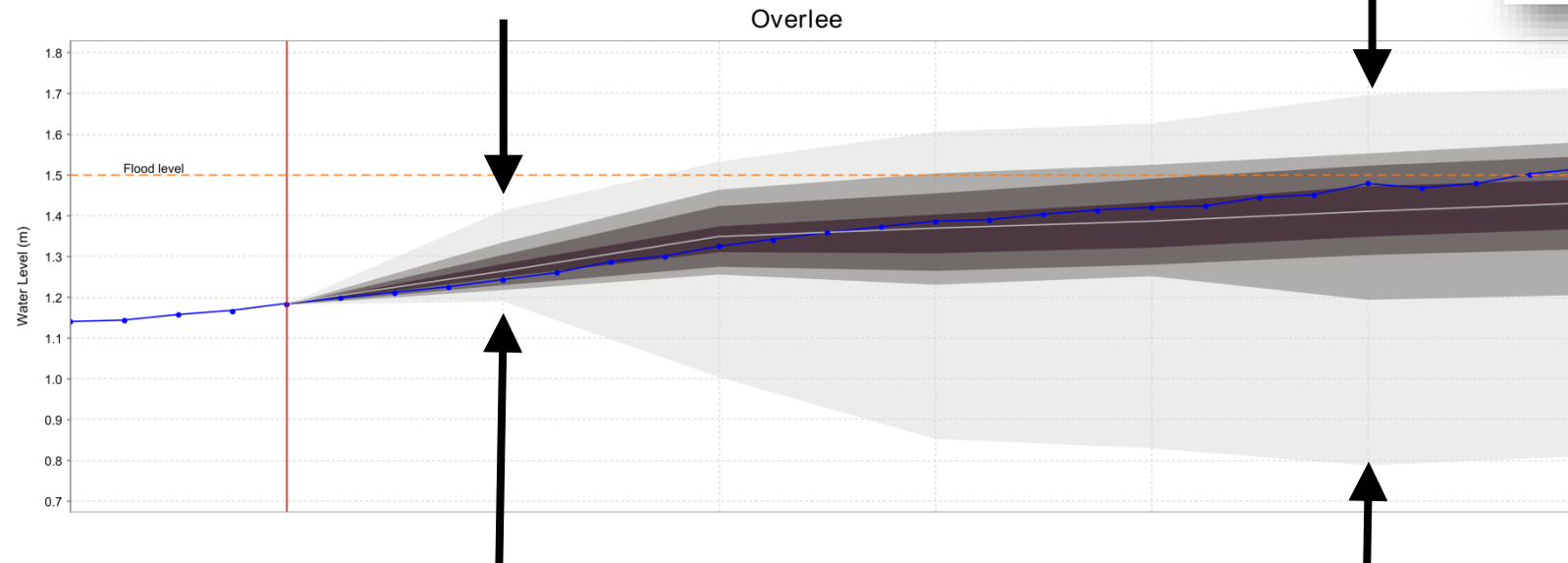
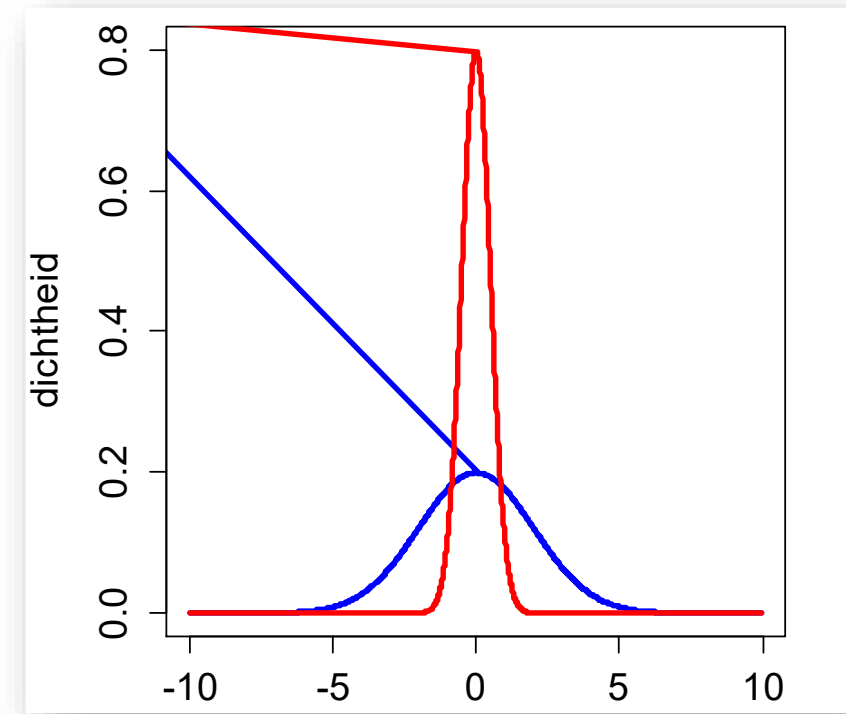
[scores tutorial](#)

Sharpness

Tendency to produce 0% and 100% probability forecasts

→ measure of the width of a predictive distribution

What is ideal? Under which conditions?



Reliability

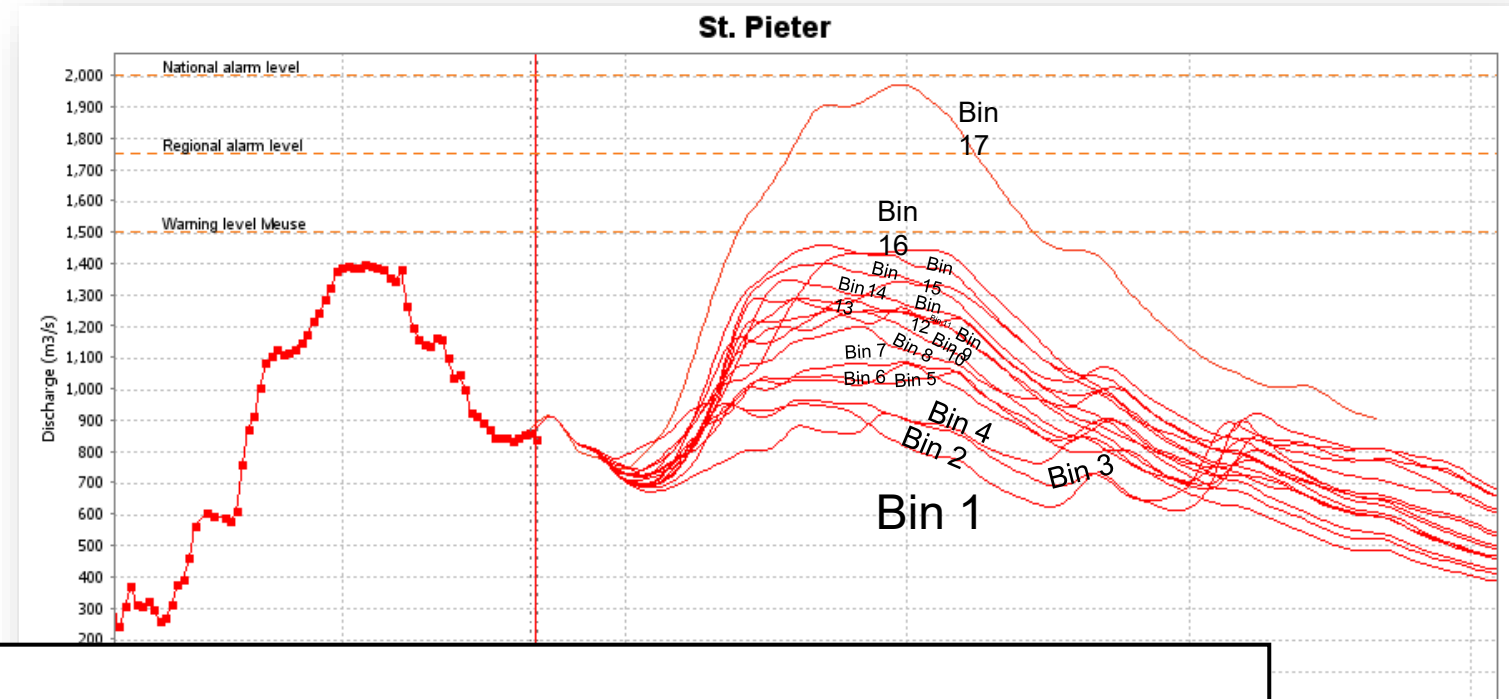
Measure with:

- **Rank Histogram (PIT diagram for probabilistic forecasts)**
- Reliability plot

Rank Histogram

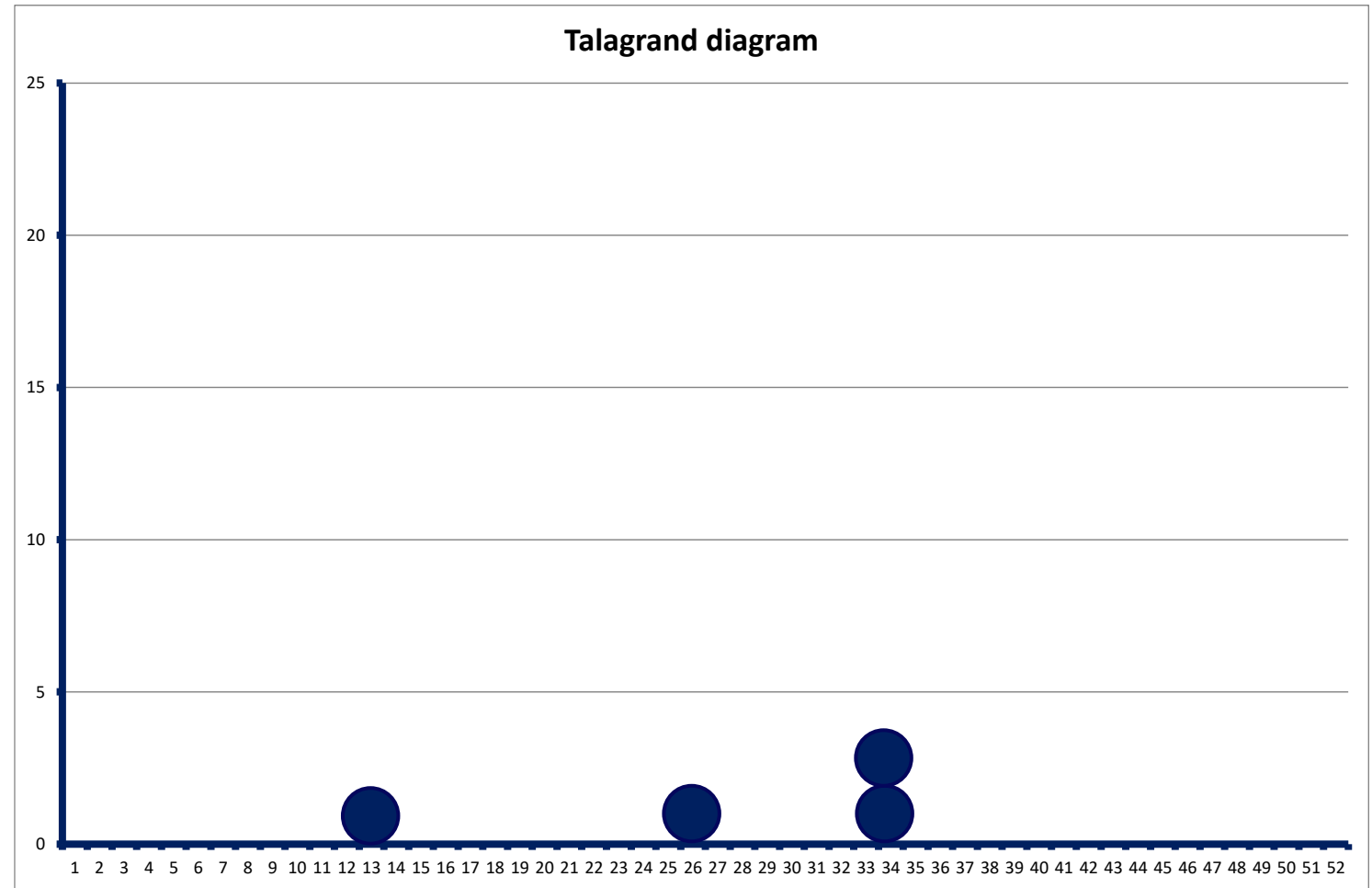
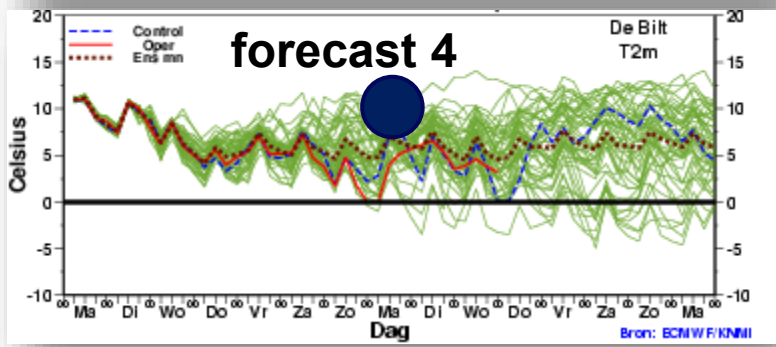
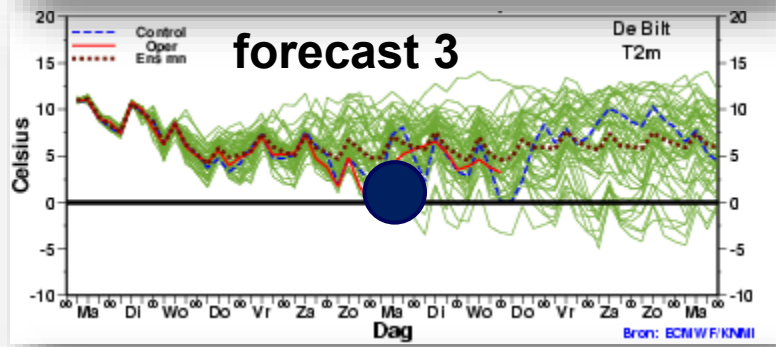
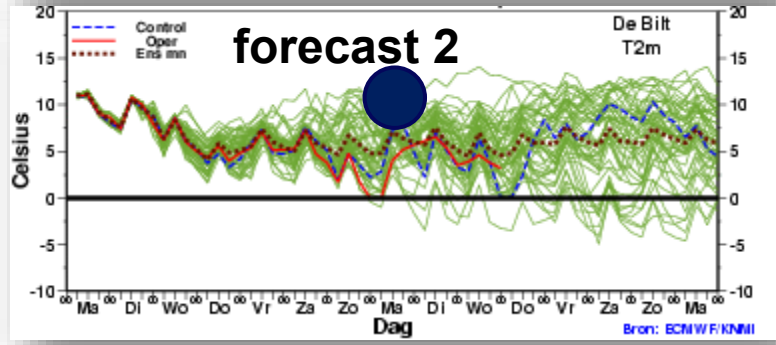
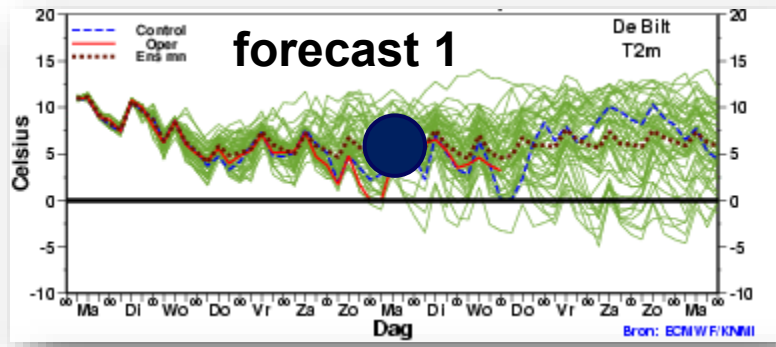
How often does the observation “land”...

- ... below the lowest member?
- ... between ...
- the lowest and second-lowest member?
- the second-lowest and third-lowest?
- the third-lowest and fourth-lowest?
- ...
- ... above the highest member?

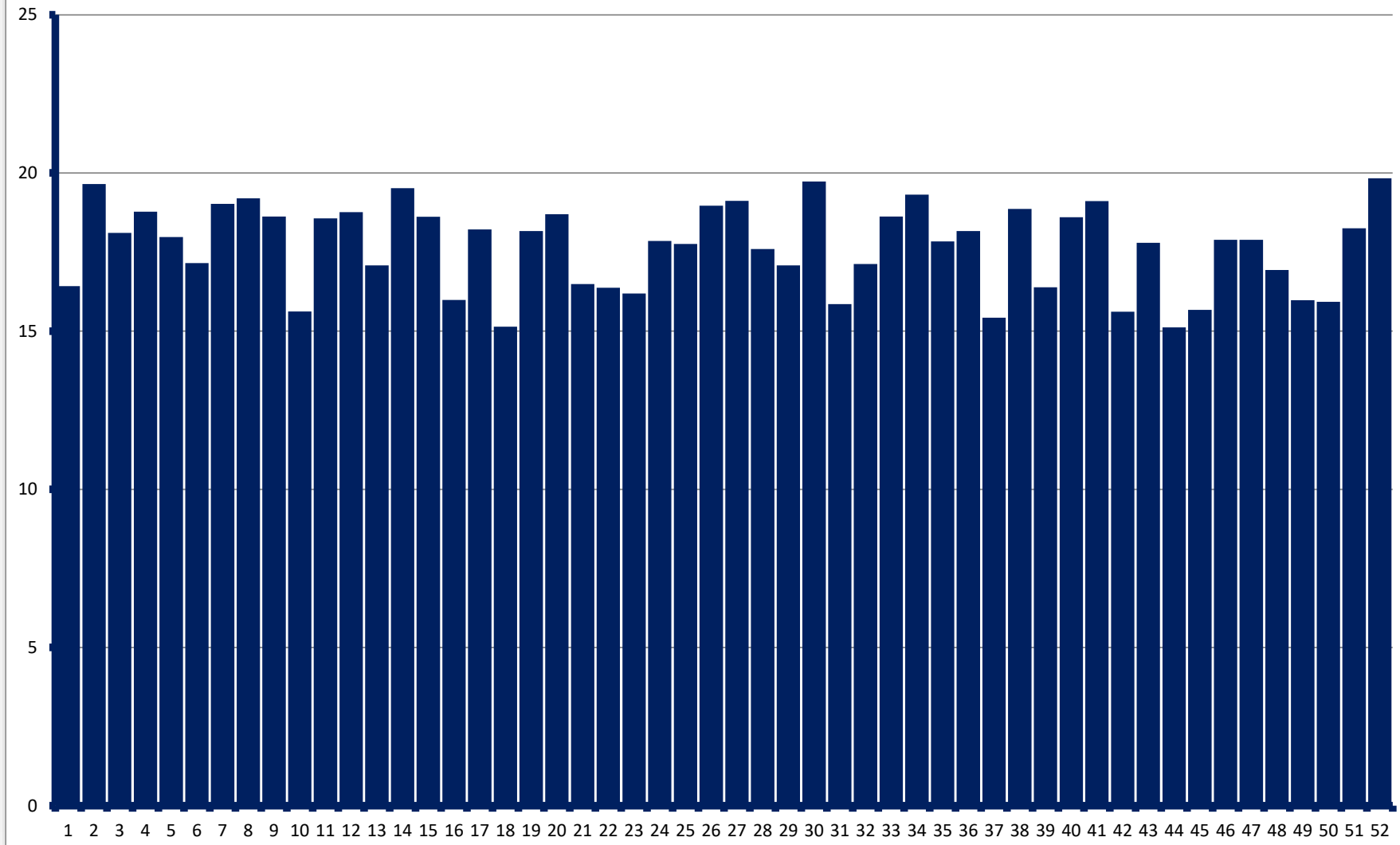


Nota Bene:

- (1) Rank histograms are valid for a specific lead time only
- (2) If your ensemble has X members then there are X+1 bins.
- (3) Bins are not necessarily of equal width

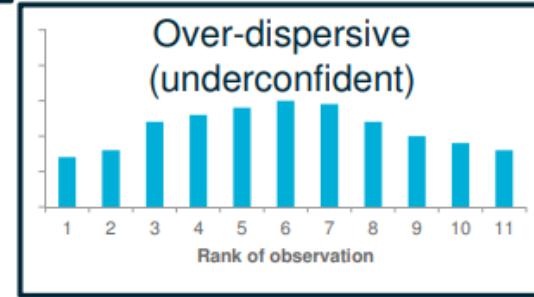
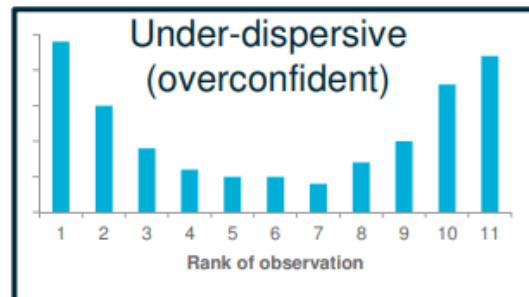
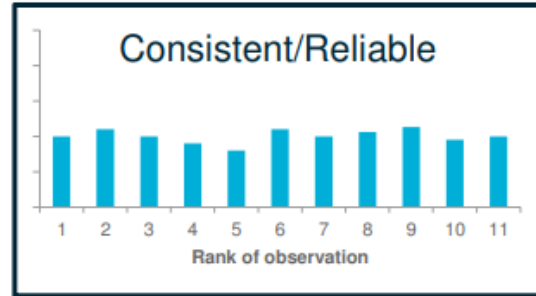
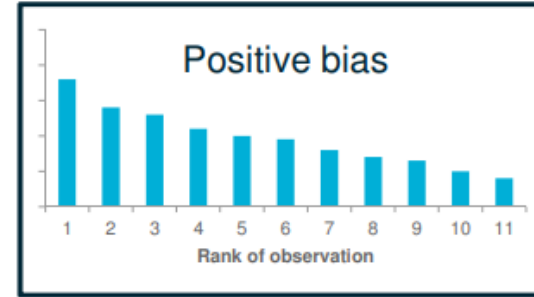
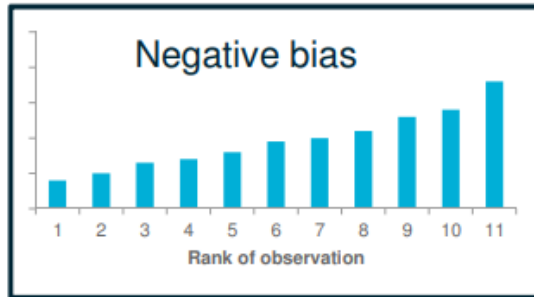


Talagrand diagram





Rank histogram

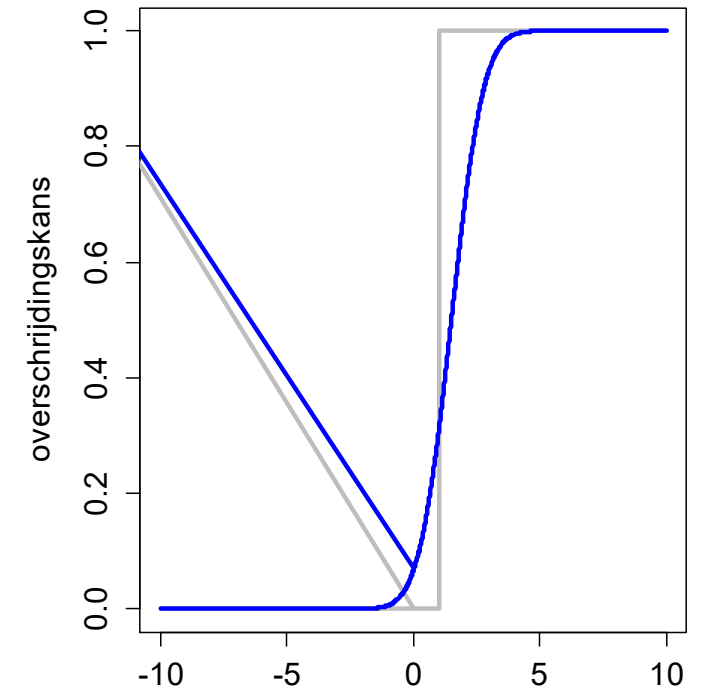
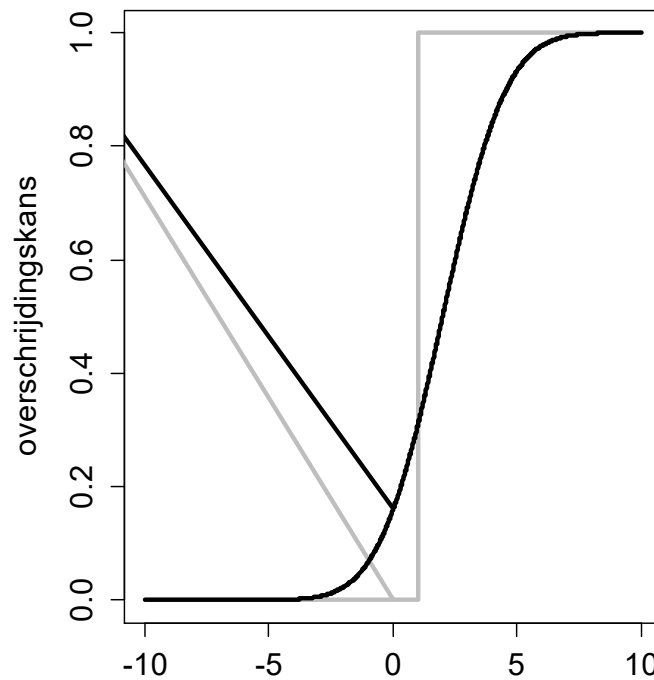


Example courtesy of Schroedter-Homscheidt et al., 2016.
(DOI: 10.13140/RG.2.2.10959.94883)

Continuous Ranked Probability Score (CRPS)

- Measure for *accuracy*, *reliability* and *sharpness*
- Area of the difference between the forecast and observed CDF
- Generalization of Mean Absolute Error for probabilistic forecasts

Simple metric for comparing forecasts
→ skill measure with CRPSS



In the operational forecasting domain, there's large potential for improving forecast verification practices

Operational dashboard for detection and analysis

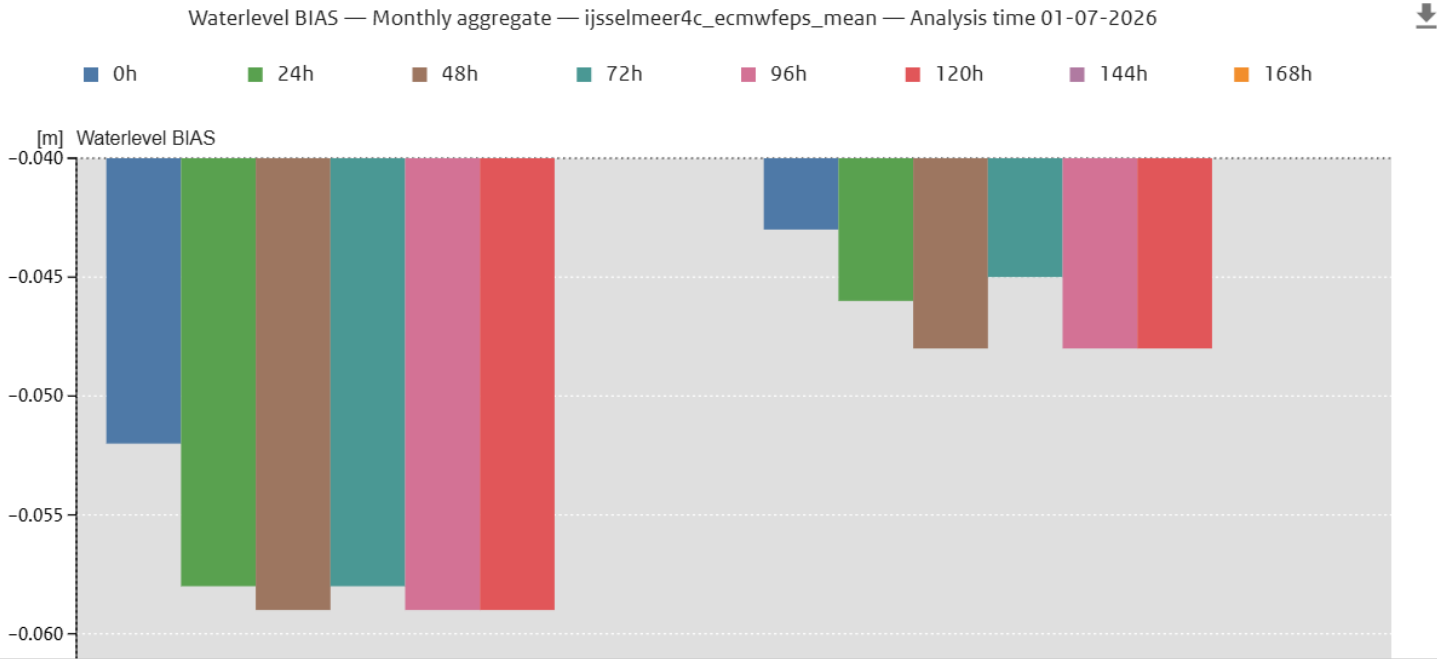
RWOS Verification Dashboard

Variable: Waterlevel | Metric: BIAS | Aggregation period: 30 days | Analysis time: 01-07-2026

Primary axis: Location | Secondary axis: Lead time

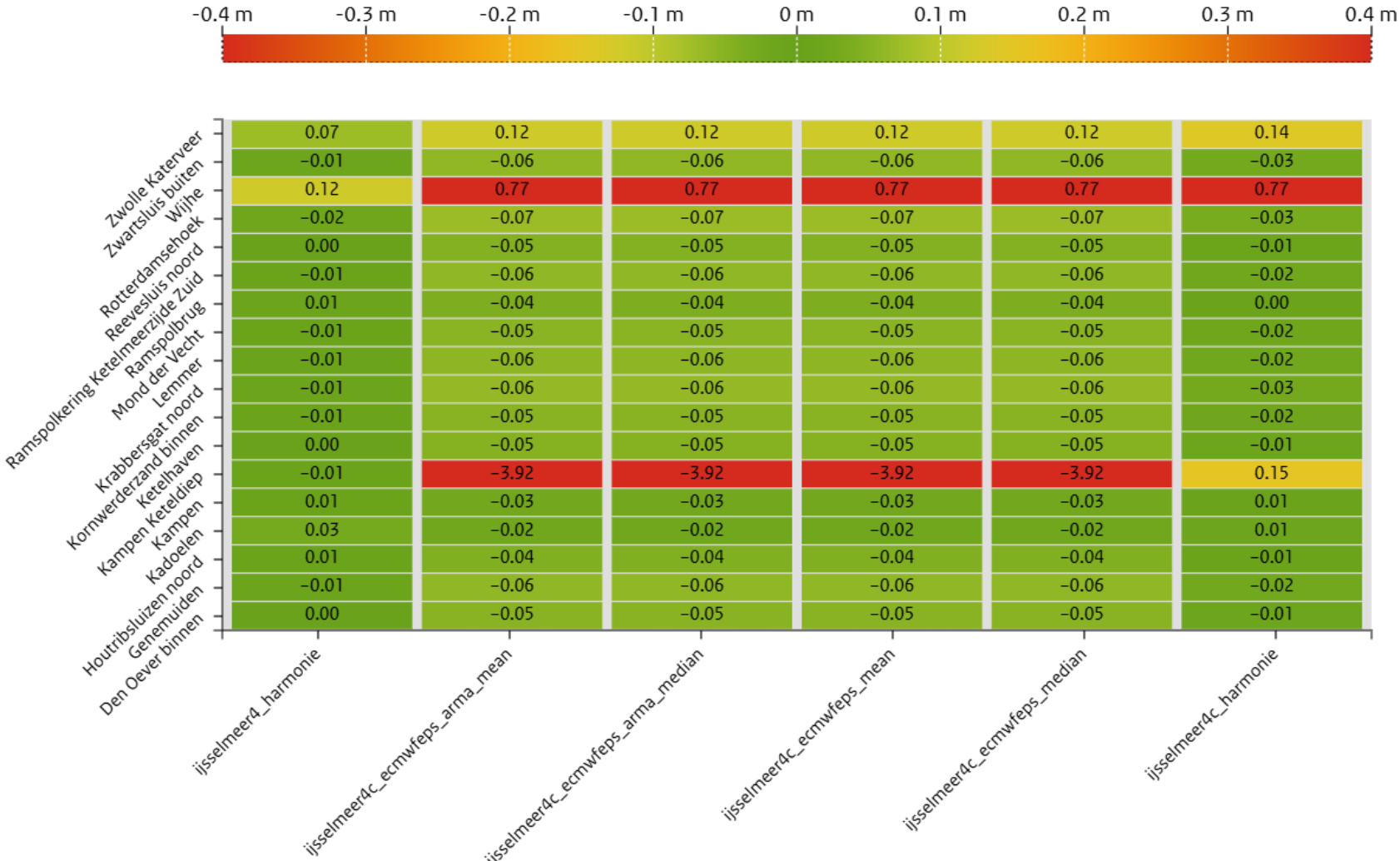
Location: Houtribsluizen noord(+1 others) | Source: ijsselmeer4c_ecmwfeps_mean | Lead time: 0h(+10 others)

BAR CHART | MATRIX



Operational dashboard for detection and analysis

Waterlevel BIAS — Monthly aggregate — Lead time 0h — Analysis time 01-07-2026



Rijkswaterstaat
Ministerie van Infrastructuur
en Waterstaat

Maturity of forecast verification at your organization?

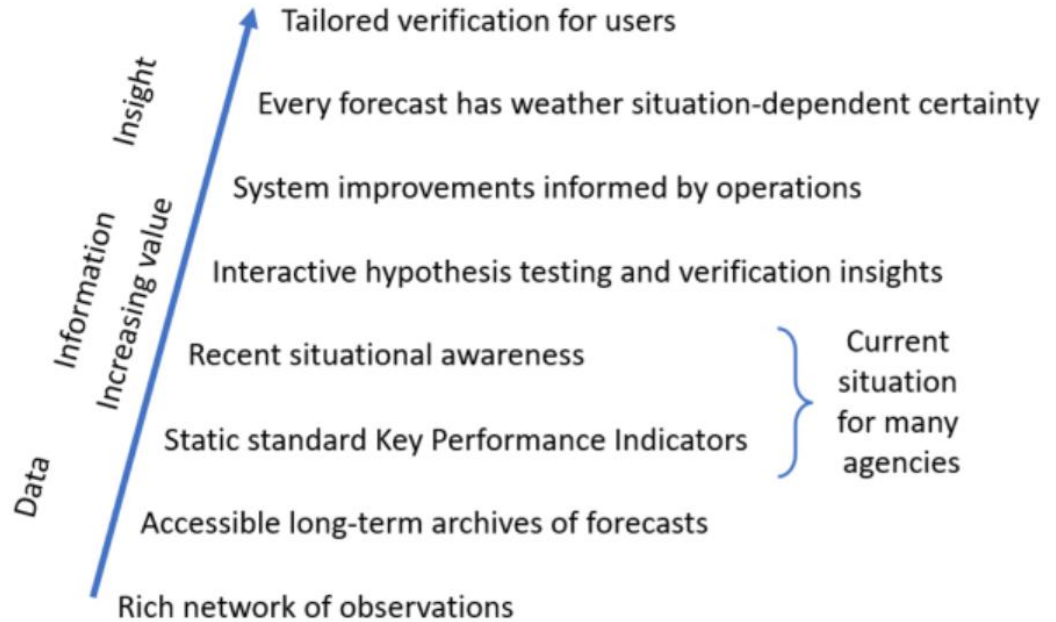
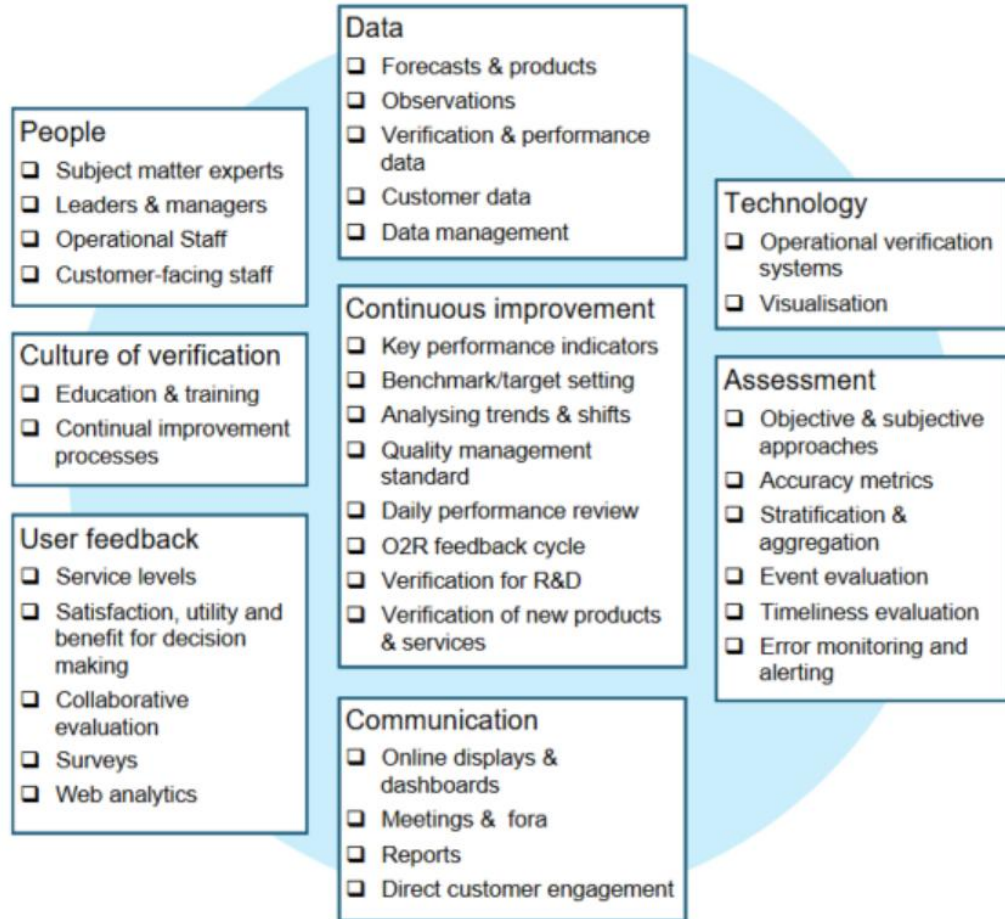


Figure 1. Verification Information Value Ladder showing the foundations for forecast verification at the bottom and increasing value of agency moves up the ladder. Reprinted from Pagano et al. (2024). with permission © American Meteorological Society.

Pagano, T.C., Ebert, E.E. and Khanarmuei, M. (2025), Enhancing Forecast Verification in National Meteorological and Hydrological Services. Meteorol Appl, 32: e70051. <https://doi.org/10.1002/met.70051>



3 Veriflow - background

Why? What is it?

Outline

Why?

What is it?

What makes it special?

Future work!

Why we started with veriflow?

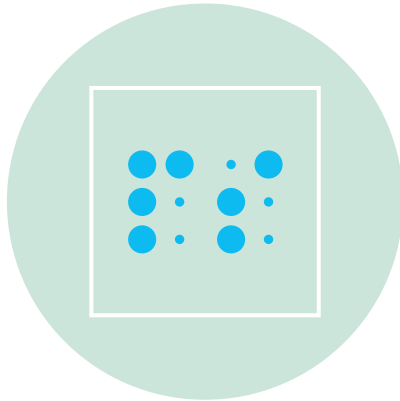
Question:

“How can we further establish solid verification practices in model development and operations?”

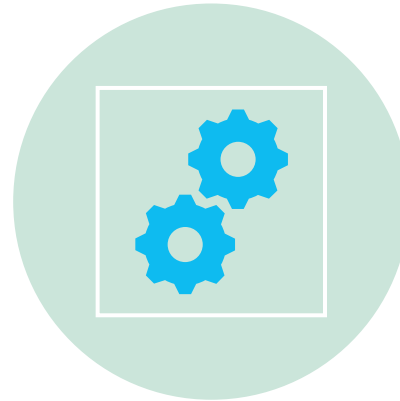
Answer:

“A robust and reproducible verification pipeline for model developers and operational forecasters”

Challenges when integrating verification in practice



Translating user-needs
to a verification strategy



Technical challenges



Organisational
integration

General technical challenges



Handling large data volumes



Transforming and matching data



Computing complex metrics

Current practices in the operational domain



Manual and fragmented

different users applied their own scripts, spreadsheets, or tools



Use-case specific

solutions built for one use-case may have limited value for others



Time-consuming

data extraction, processing, and comparison required many manual steps



Hard to reproduce

methods and assumptions weren't consistently documented

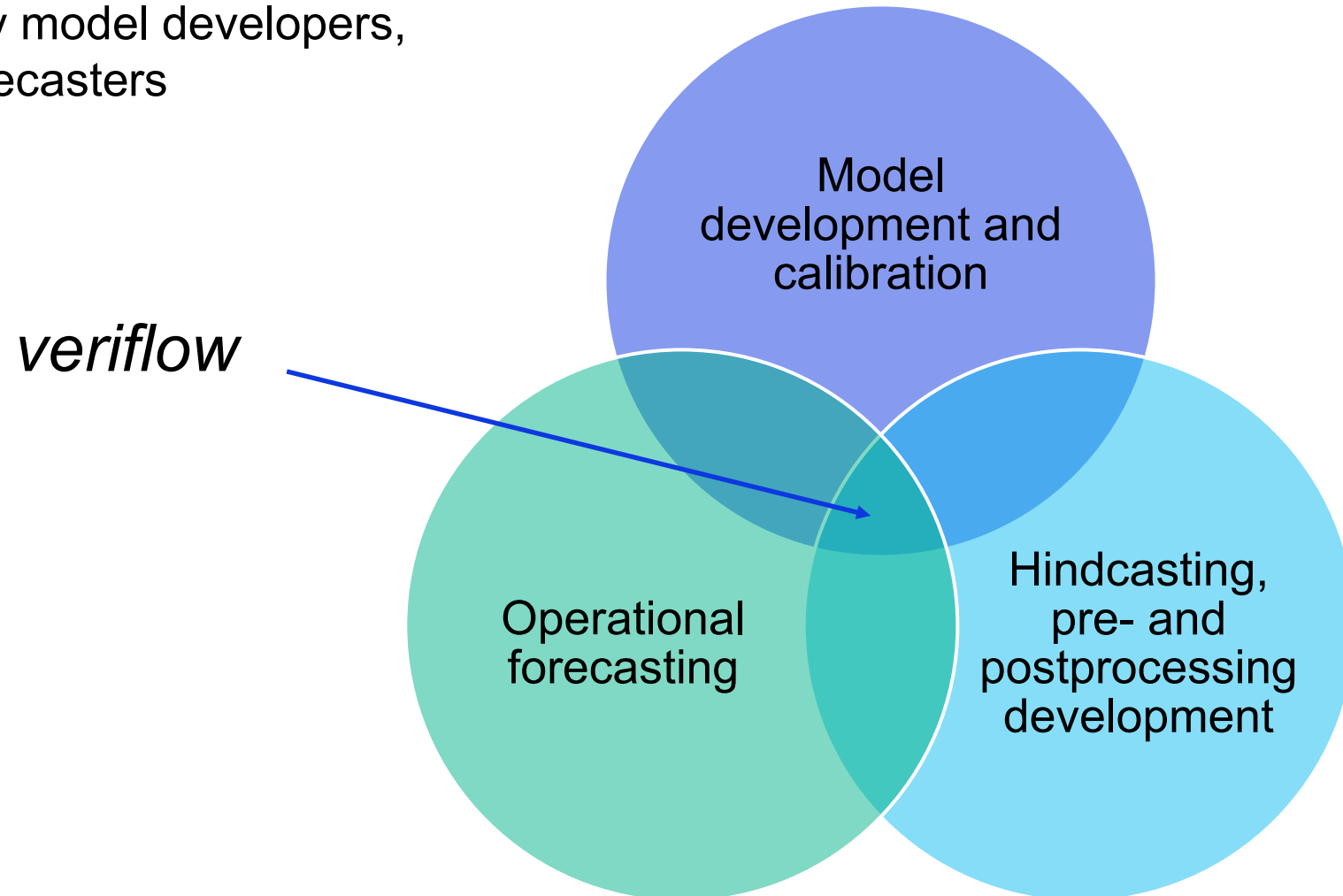
So we started developing *veriflow*

To enable Delft-FEWS users to perform **verification and skill assessment** of operational forecasts in an **efficient, robust and reproducible way** — for all types of forecasts (deterministic, ensemble, and probabilistic)



And to standardize verification practices between research, development and operational application

So it can be used by model developers, researchers and forecasters



The story of *veriflow*

Deltares started development in 2024
Collaboration with Rijkswaterstaat

In collaboration with Vortech (scientific software engineers)

- Code review
- Architecture

A Python package

It's open source (MIT)

Tested and documented

- Docs: <https://deltares.github.io/veriflow/>
- GitHub: <https://github.com/Deltares/veriflow>



README Contributing MIT license

pypi v0.4.1 docs latest codecov 82% CI failing docs build passing

Veriflow

A reproducible verification pipeline for evaluating model outputs and forecasts.

- Fetching data
- Computing scores
- Writing results

[Schematic overview of veriflow pipeline](#)

Key features

- Full control over the verification pipeline via configuration
- Native integration with [Delft-FEWS](#)
- Native integration with local and remote (S3) [Zarr](#)
- Builds on [scores](#) for computation of scores. This package has extensive functionality, and its documentation is world-class.
- Extensible with your own (private) datasources, scores and datasinks
- Optimized internal datamodel for efficient computation

We make use of *scores* for computation of metrics

- Developed by the BoM (Bureau of Meteorology, Australia)
- Supports continuous, categorical, probabilistic and spatial verification scores

- World-class documentation and tutorials
- Active development
- Fast computation
- R&D on new scores

scores: Verification and Evaluation for Forecasts and Models

JOSS 10.21105/joss.06889 CodeQL passing coverage 100% launch binder pypi v2.5.0 conda-forge v2.5.0

A list of over 75 metrics, statistical techniques and data processing tools contained in `scores` is [available here](#).

`scores` is a Python package containing mathematical functions for the verification, evaluation and optimisation of forecasts, predictions or models. It supports labelled n-dimensional (multidimensional) data, which is used in many scientific fields and in machine learning. At present, `scores` primarily supports the geoscience communities; in particular, the meteorological, climatological and oceanographic communities.

Leeuwenburg, T., Loveday, N., Ebert, E. E., Cook, H., Khanarmuei, M., Taggart, R. J., Ramanathan, N., Carroll, M., Chong, S., Griffiths, A., & Sharples, J. (2024). *scores*: A Python package for verifying and evaluating models and predictions with xarray. *Journal of Open Source Software*, 9(99), 6889. <https://doi.org/10.21105/joss.06889>

The scores documentation is world-class

4 important categories of verification scores exist

- Continuous
- Probability
- Categorical
- Spatial

Overview

Below is a **curated selection** of the metrics, tools and statistical tests included in [scores](#). ([Click here for the full list.](#))

	Description	Selection of Included Functions
Continuous	Scores for evaluating single-valued continuous forecasts.	E.g. MAE, MSE, RMSE, Bias, Pearson's Correlation Coefficient, Kling-Gupta Efficiency, NSE, Flip-Flop Index, Quantile Loss, Quantile Interval Score, Interval Score, and threshold weighted scores for expectiles, quantiles and Huber Loss. See all.
Probability	Scores for evaluating forecasts that are expressed as predictive distributions, ensembles, and probabilities of binary events.	E.g. Brier Score, PIT, CRPS for CDFs and ensembles (including threshold weighted versions), and Isotonic Regression (reliability diagrams). See all.
Categorical	Scores for evaluating forecasts of categories.	E.g. 18 binary contingency table (confusion matrix) metrics, the Fixed Risk Multicategorical (FIRM) Score, the SEEPS score and the Risk Matrix Score. See all.
Spatial	Scores that take into account spatial structure.	Fractions Skill Score. See all.

Scores tutorials

Probability

Probabilistic forecasts of binary events can be expressed with values between 0 and 1, and observations are exactly 0 (event did not occur), or 1 (event occurred). The metric is then calculated the same way as MSE and is defined as

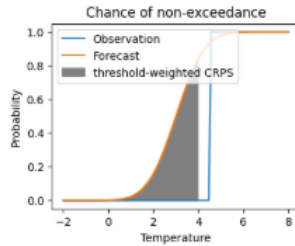
$$s(x, y) = (x - y)^2$$

Where

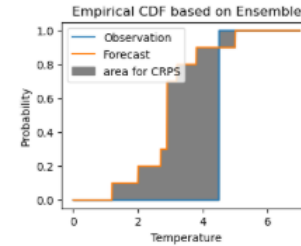
- x is the forecast between 0 and 1, and
- y is the observation which is either 0 or 1.

The Brier score is a strictly proper scoring rule where lower values are better (it is negatively oriented), where a perfect score is 0 and the worst score is 1.

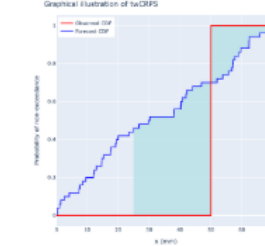
Brier Score



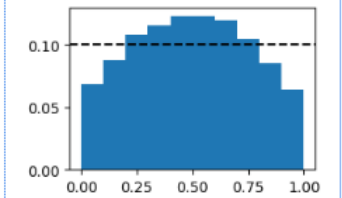
CRPS for Forecasts Expressed as CDFs



CRPS for Ensemble Forecasts

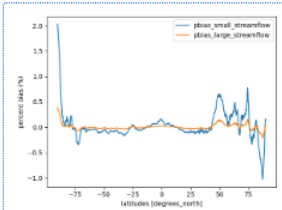


twCRPS for Ensemble Forecasts

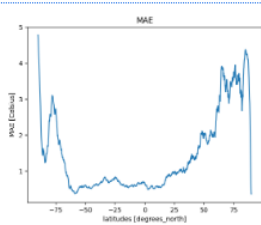


Probability Integral Transform (PIT)

Continuous



Additive Bias, Multiplicative Bias and Percent Bias



MAE

From Nash and Sutcliffe (1970), standardized to the version in (2017), NSE is defined as follows:

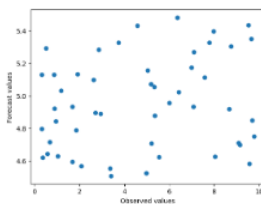
$$NSE = 1 - \frac{\sum_{t=1}^n (f_t - o_t)^2}{\sum_{t=1}^n (o_t - \bar{o})^2} = 1 - \frac{MSE}{\sigma_o^2}$$

Where

- f_t is the forecast or predicted value
- o_t is the observed value
- \bar{o} is the mean of the observed values
- σ_o^2 is the variance of the observed values
- MSE is the mean squared error
- σ_o^2 is the mean squared error, equivalent to the variance of the observed values

Source also supports the use of (2017). These can be used to scale both the individual forecast error in the numerator and the deviation from the observation mean in the denominator. This is also known as the weighted NSE or wNSE. For an application of this, refer to (2017) and (2018).

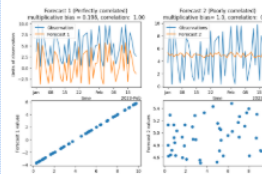
Nash-Sutcliffe Model Efficiency Coefficient (NSE)



Pearson's Correlation



Spearman's Correlation



Kling-Gupta Efficiency

The quantile loss (quantile score) is a combined scoring function for an q -quantile forecast, where $q \in (0, 1)$. The quantile loss function is also used in adaptive learning, in quantile regression, which focuses on estimating different quantiles of the conditional distribution of the response variable. In finance, quantiles are often referred to as value-at-risk.

The quantile loss function is defined as:

$$Q(x, y) = \begin{cases} (x - y) & \text{if } x \geq y \\ (y - x) & \text{if } x < y \end{cases}$$

where Q is the scoring function (Brier quantile loss), x and y are forecast and observation, respectively.

Lower values of the quantile loss are better.

More information about the quantile loss function can be found in this article:

- Grubis, T. (2015). Making and evaluating point forecasts. *Journal of the American Statistical Association*, 106, 740-750. <https://doi.org/10.1080/01621459.2015.1033388>

Quantile Loss

https://scores.readthedocs.io/en/stable/tutorials/Tutorial_Gallery.html

Easy use in Jupyter Notebooks (stand alone or server)

Inspirations from BoM's Jive system

(Loveday et al., 2024)

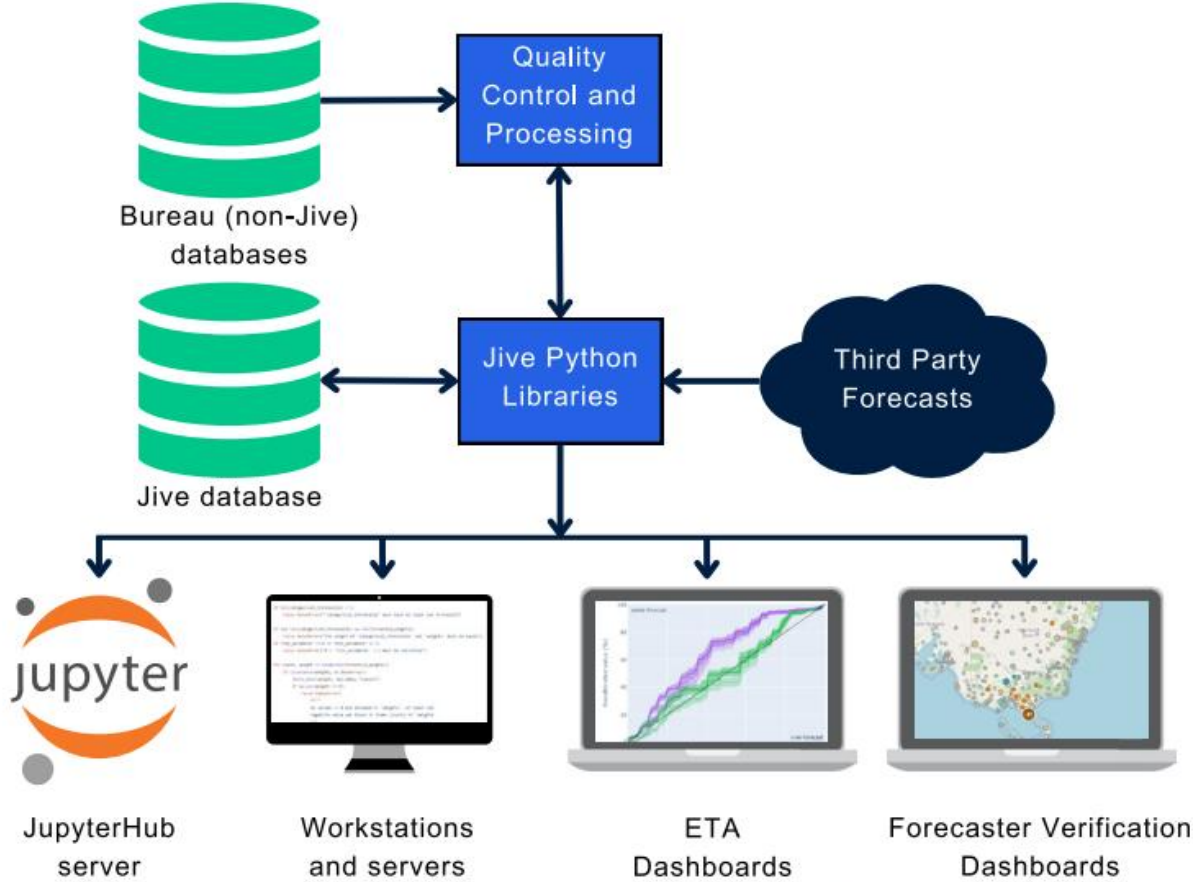


FIG. 2. Schematic diagram of the Jive system. Bureau databases supply forecasts and raw observations. Integrated Jive Python libraries do everything from data retrieval, processing, and storage, as well as generate statistics and visualize outputs on dashboards.

So what can *veriflow* do for you?

- Veriflow provides you with **flexible and reproducible verification pipeline**
- **Lowers the technical barrier** for verification
- Frees up experts to focus on **interpreting results** instead of managing data
- Builds trust and accountability by making verification **configurable and reproducible**
- Enables **consistent and comparable** verification across models, locations and times

Just like Delft-FEWS, veriflow is a community-driven effort...

We welcome contributions and hope to work together to further establish good verification practices in the operational domain

Veriflow is a reproducible verification pipeline tool

Connects to any Delft-FEWS system for data retrieval

Automates the verification workflow — from data retrieval to computation to result storage

Uses a simple configuration — specify what to verify and how

Supports all forecast types — deterministic, ensemble, and probabilistic

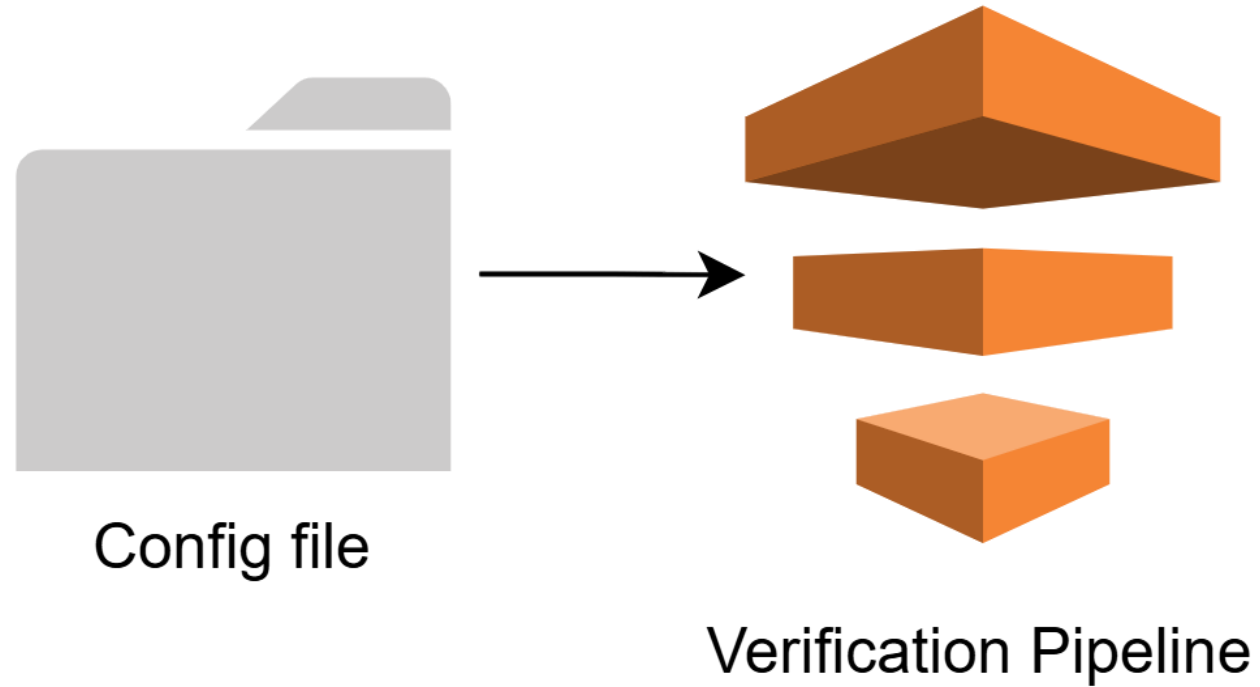
Has flexible inputs — allows you to connect to a private database

Has flexible scores — allows you implement your own custom scores

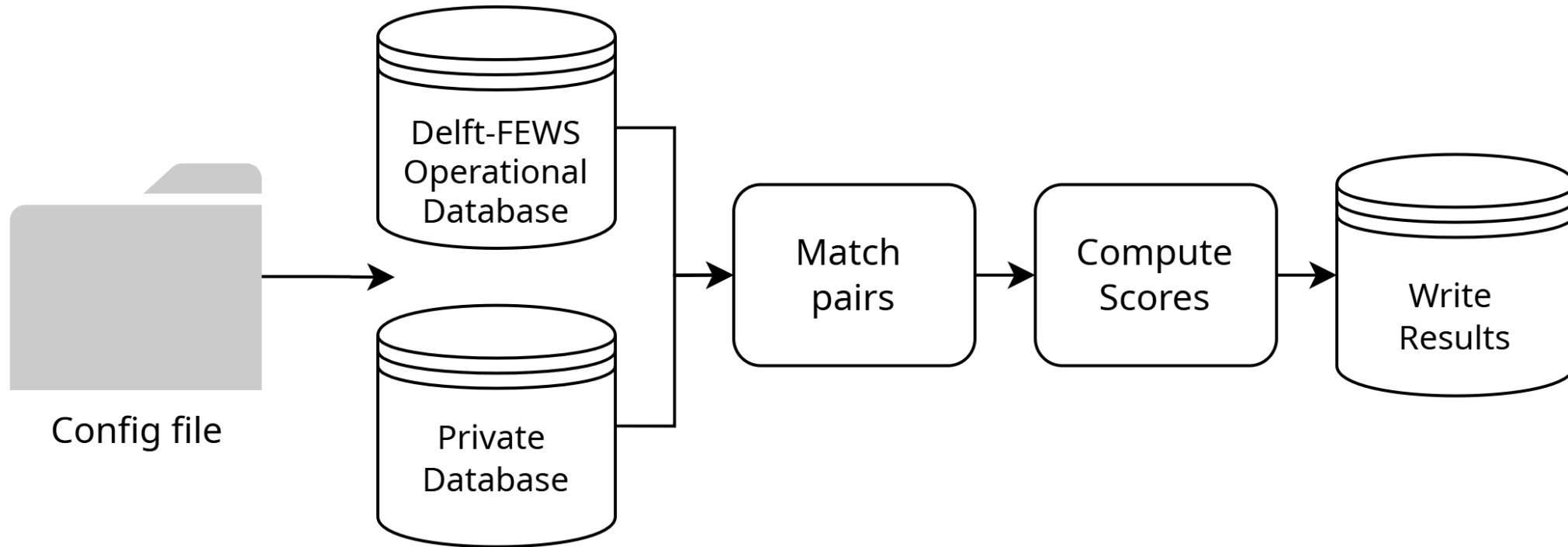
Has flexible outputs — results can be visualized

- In custom dashboards
- In Delft-FEWS (planned 2026)
- In any other format

Alright, so how does it work **exactly**?



Alright, so how does it work exactly?

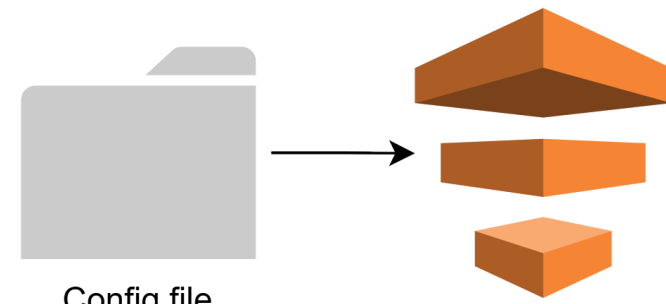


We define the verification pipeline in a simple config file

```
ConfigSchema (config.schema.json) | DaveCasson, 6 hours ago | 2 authors (You and one other)
1 # yaml-language-server: $schema=https://deltares.github.io/veriflow/v0/config.schema.json
2 version: v0
3 general:
4   verification_period:
5     dimension: forecast_reference_time
6     start: "2025-05-20T00:00:00Z"
7     end: "2026-06-01T00:00:00Z"
8   verification_pairs:
9     - id: QR_GEPS
10       obs: observed
11       sim: simulated_GEPS
12       variable: discharge
13     - id: QR_GEFS
14       obs: observed
15       sim: simulated_GEFS
16       variable: discharge
17     - id: QR_IFS
18       obs: observed
19       sim: simulated_IFS
20       variable: discharge
21   lead_times:
22     unit: D
23     values: [1, 2, 3, 4, 5, 6]
24 > datasources: ...
57 scores:
58   - score_adapter: crps_for_ensemble
59   - score_adapter: rank_histogram
70   reduce_dims:
71     - forecast_reference_time
```

In just 70 lines of YAML config,
We define

- What period to verify
- What models to verify
- What lead times to verify
- Where to get the data (Delft-FEWS)
- What scores to compute



Config file

Verification Pipeline

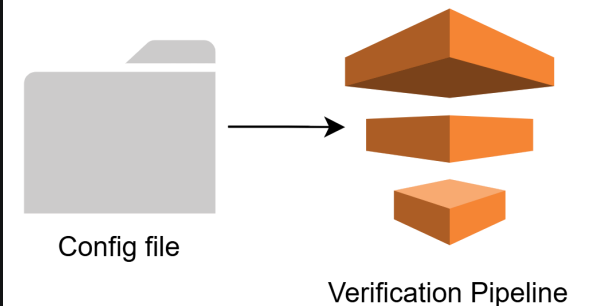
We run the *veriflow* pipeline in just one line of code

In just one line of code, we run the *veriflow* pipeline

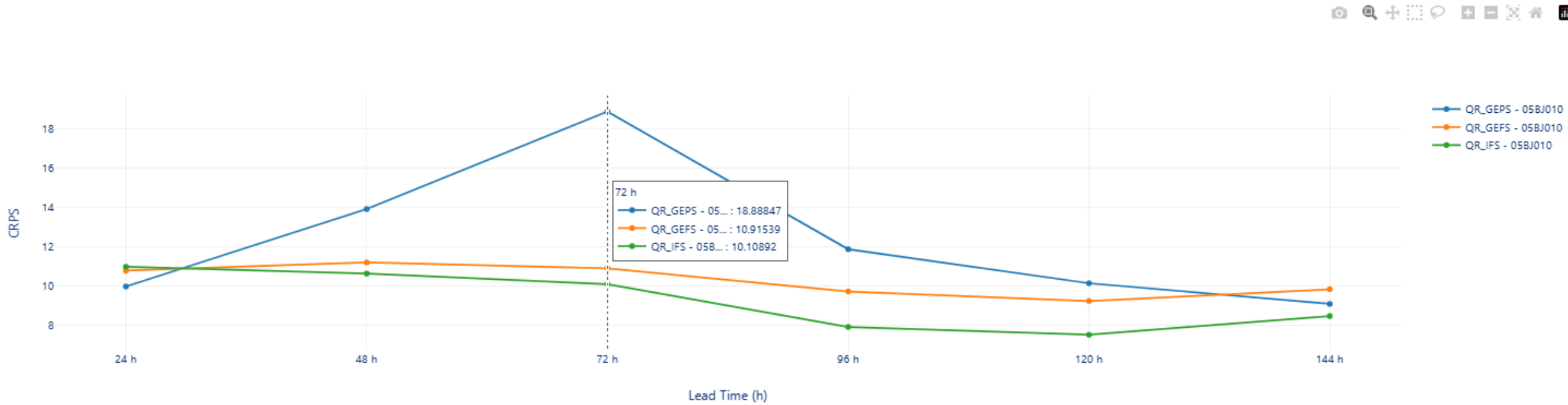
Running the *veriflow* pipeline

```
config_file_path = base_config / "Raven_Elbow_Discharge_Ensemble.yaml"  
ods = run_pipeline(config=(config_file_path, "yaml"))
```

```
2026-06-12 16:38:53,599 - INFO - Successfully initialized the configuration.  
    verification_period_start = 2025-05-20 00:00:00  
    verification_period_end = 2026-06-01 00:00:00  
2026-06-12 16:38:53,603 - INFO - Start getting data from FewsWebservice.  
2026-06-12 16:38:54,109 - INFO - Successfully got data from FewsWebservice.  
2026-06-12 16:38:54,110 - INFO - Start getting data from FewsWebservice.  
2026-06-12 16:38:54,660 - INFO - Successfully got data from FewsWebservice.  
2026-06-12 16:38:54,660 - INFO - Start getting data from FewsWebservice.  
2026-06-12 16:38:55,203 - INFO - Successfully got data from FewsWebservice.  
2026-06-12 16:38:55,203 - INFO - Start getting data from FewsWebservice.  
2026-06-12 16:38:55,794 - INFO - Successfully got data from FewsWebservice.  
2026-06-12 16:38:55,797 - INFO - Successfully loaded all data from sources.  
2026-06-12 16:38:55,831 - INFO - Successfully computed CrpsForEnsemble for verification pair QR_GEPS.  
2026-06-12 16:38:55,853 - INFO - Successfully computed CrpsForEnsemble for verification pair QR_GEFS.  
2026-06-12 16:38:55,880 - INFO - Successfully computed CrpsForEnsemble for verification pair QR_IFS.  
2026-06-12 16:38:55,898 - INFO - Successfully computed RankHistogram for verification pair QR_GEPS.  
2026-06-12 16:38:55,916 - INFO - Successfully computed RankHistogram for verification pair QR_GEFS.  
2026-06-12 16:38:55,935 - INFO - Successfully computed RankHistogram for verification pair QR_IFS.  
2026-06-12 16:38:55,982 - INFO - Successfully wrote results of verification pair QR_GEPS to CFCompliantNetCDF.  
2026-06-12 16:38:56,019 - INFO - Successfully wrote results of verification pair QR_GEFS to CFCompliantNetCDF.  
2026-06-12 16:38:56,061 - INFO - Successfully wrote results of verification pair QR_IFS to CFCompliantNetCDF.  
2026-06-12 16:38:56,062 - INFO - Verification pipeline completed successfully.
```



An example of CRPS results for the 3 NWP products at various lead times



So *veriflow* can be easily applied to Delft-FEWS

Lowering the technical barrier for verification

Future

2026

- Streamline filtering:
 - a. In time
 - b. Co-variables
 - c. According to climatology
 - d. Thresholds
- Skills scores
- More on detection: e.g. hit and false alarm rates
- Further integration in real-time FEWS
- 2026: Design and testing, more development in 2027
- Dashboards stand alone and FEWS



4 Veriflow – trying it out

Exercises from demo gallery

Tutorial gallery:

<https://deltares.github.io/veriflow/gallery.html>

Gallery

Explore *veriflow* through hands-on, case-based tutorials. Each case bundles a set of Jupyter notebooks that walk through a complete verification workflow, from accessing data to computing scores and interpreting the results.

Every notebook can be **downloaded** or **run live in Binder** – look for the banner at the top of each notebook page.



The Delft-FEWS (OpenFEWS) Case

A step-by-step tutorial built around a real flood event in the Elbow Watershed (Alberta, Canada). Verify precip



The Rhine Case

A compact, end-to-end example comparing multiple models for the Rhine by running one integrated verification pipeline. Learn the core *veriflow* workflow and how to interpret the resulting verification metrics.

Hindcast,
Deterministic
Ensemble
FEWS webservice

Raw and
Postprocessed Precip
Ensemble
ZARR archive

The Rhine case

A compact, end-to-end example that runs a single integrated verification pipeline to compare multiple models for the Rhine. Start with the **basics** to learn the core *veriflow* workflow, then move on to **interpretation** to make sense of the resulting verification metrics.

Each notebook page includes a banner to **download** it or **open it in Binder**.



1a • Basics

Configure a verification experiment, run the *veriflow* pipeline, inspect the returned `OutputDataset`, and explore results with the interactive app.



1b • Interpretation

Interpret the verification results: how forecast quality changes with lead time, how well the ensemble captures uncertainty, and the effect of post-processing.

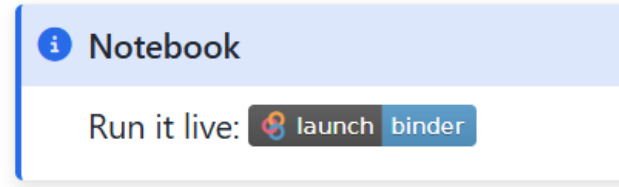
How will you do this?

View

Just look at website

See the notebook and results

Browser with Binder



Private session

Closed after inactive for
5-10min

Local installation

Info at: <https://github.com/Deltares/veriflow>

Python ≥ 3.11

Install and download tutorial
separately, or clone the repo.

Installation

Install from PyPI:

```
pip install veriflow
```

Or using [uv](#):

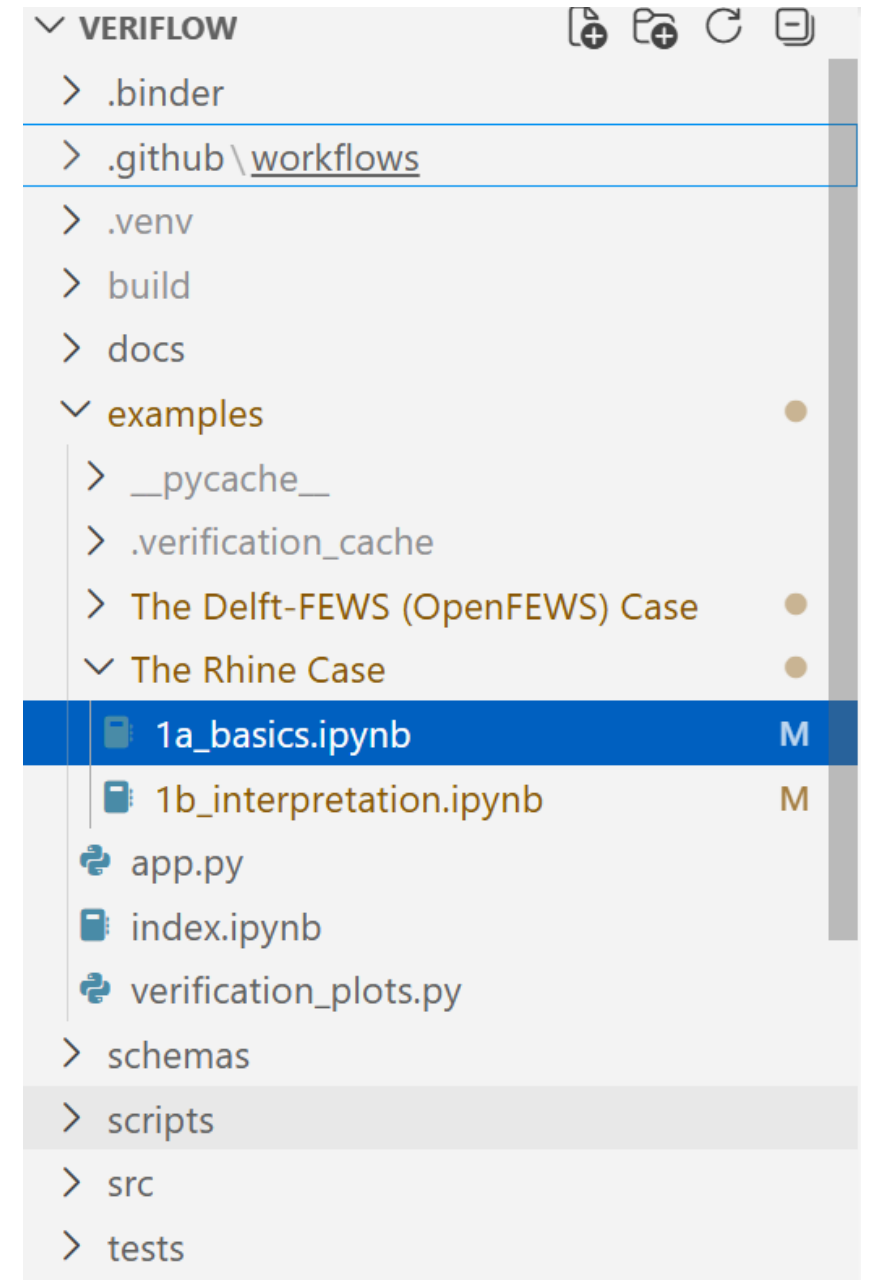
```
uv pip install veriflow
```

See [CONTRIBUTING.md](#) for development setup.

The code

Rhine tutorial notebook uses matplotlib for plotting

```
pip install matplotlib
```



The verification period “dimension”

- The “dimension” in the `verification_period` allows you to define the dimension to use when defining your “verification period”
- Time:
 - define start and end along “time”
 - → used for historical simulations
 - → when used with forecasts, will mask all forecast values outside a given window
- `Forecast_reference_time`:
 - define start and end along “forecast_reference_time”
 - → standard for forecast verification: uses complete forecasts with a `forecast_reference_time` between start and stop

```
1 # yaml-language-server: $schema=https://deltares.gitl
2 version: v0
3 general:
4   verification_period:
5     dimension: time
6     start: "2026-05-15T00:00:00Z"
7     end: "2026-06-08T00:00:00Z"
```

The verification pair

- The verification pair defines allows you to define what data to compare.
 - id: used as metadata in output
 - obs → reference your observed datasource by it's "source" property
 - sim → same concept as obs
 - variable: the variable to select from the datasource

```
! Elbow_Renanalysis_Precip.yaml •
examples > The Delft-FEWS (OpenFEWS) Case > config > ! Elbow_Renanalysis
You, 5 days ago | 2 authors (You and one other) | ConfigSchema (config.sche
# yaml-language-server: $schema=https://deltares.git
1  version: v0
2  general:
3    verification_period:
4      dimension: time
5      start: "2026-05-15T00:00:00Z"
6      end: "2026-06-08T00:00:00Z"
7    verification_pairs:
8      - id: PC_RDPA
9        obs: observed
10       sim: simulated_RDPA
11       variable: PC
12      - id: PC_HRDPA
13        obs: observed
14        sim: simulated_HRDPA
15        variable: PC
16  datasources:
17    - import_adapter: fewswbservice
18      source: observed
19      data_type: observed_historical
20      location_ids: ...
21  > parameter_ids:
22    - PC.obs
23  module_instance_id: ImportMeteoStations
24  webservice_version: "2025.02"
25  You, 6 days ago • wip
26  - import_adapter: fewswbservice
27    source: simulated_RDPA
```

Tips for new configurators

1

Use YAML language support to validate config against a schema

2

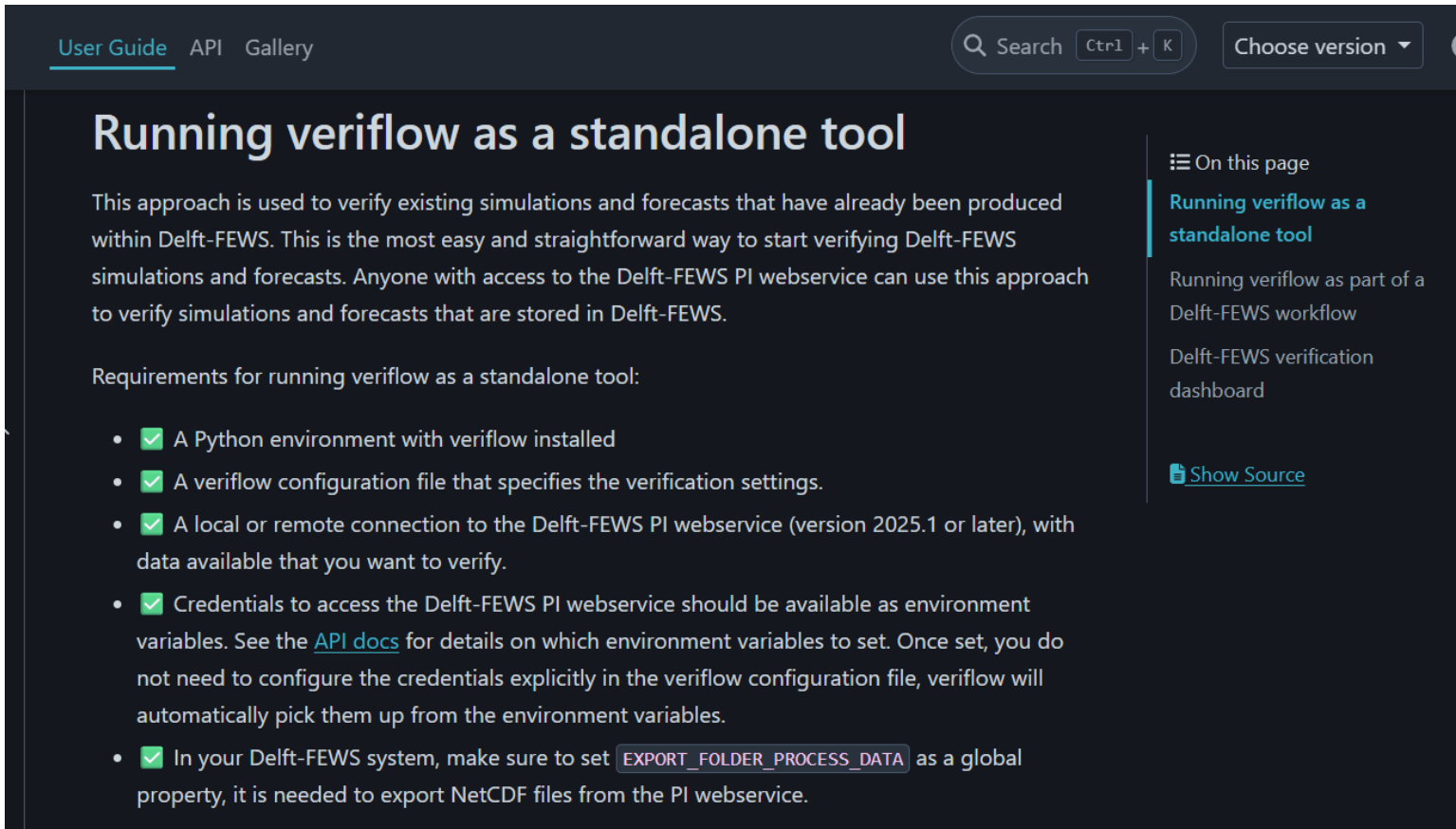
Use YAML language support to get useful hints on what is expected in certain fields

3

Use the documentation pages for concepts behind configuration

References

https://deltares.github.io/veriflow/user_guide/best_practices/delft_fews.html



The screenshot shows a dark-themed web page for the Veriflow user guide. At the top, there are navigation links for 'User Guide', 'API', and 'Gallery'. A search bar contains the text 'Search' and a keyboard shortcut 'Ctrl + K'. To the right is a 'Choose version' dropdown menu. The main heading is 'Running veriflow as a standalone tool'. Below the heading is a paragraph explaining that this approach is used to verify existing simulations and forecasts produced within Delft-FEWS. A section titled 'Requirements for running veriflow as a standalone tool:' follows, containing a bulleted list of five items, each with a green checkmark icon. The list includes requirements for a Python environment, a configuration file, a connection to the Delft-FEWS PI webservice, credentials, and a specific environment variable. On the right side, there is a 'On this page' section with a list of links: 'Running veriflow as a standalone tool' (highlighted), 'Running veriflow as part of a Delft-FEWS workflow', and 'Delft-FEWS verification dashboard'. At the bottom of this section is a 'Show Source' link.

User Guide API Gallery

Search Ctrl + K Choose version

Running veriflow as a standalone tool

This approach is used to verify existing simulations and forecasts that have already been produced within Delft-FEWS. This is the most easy and straightforward way to start verifying Delft-FEWS simulations and forecasts. Anyone with access to the Delft-FEWS PI webservice can use this approach to verify simulations and forecasts that are stored in Delft-FEWS.

Requirements for running veriflow as a standalone tool:

- ✓ A Python environment with veriflow installed
- ✓ A veriflow configuration file that specifies the verification settings.
- ✓ A local or remote connection to the Delft-FEWS PI webservice (version 2025.1 or later), with data available that you want to verify.
- ✓ Credentials to access the Delft-FEWS PI webservice should be available as environment variables. See the [API docs](#) for details on which environment variables to set. Once set, you do not need to configure the credentials explicitly in the veriflow configuration file, veriflow will automatically pick them up from the environment variables.
- ✓ In your Delft-FEWS system, make sure to set `EXPORT_FOLDER_PROCESS_DATA` as a global property, it is needed to export NetCDF files from the PI webservice.

On this page

- Running veriflow as a standalone tool
- Running veriflow as part of a Delft-FEWS workflow
- Delft-FEWS verification dashboard

Show Source

Useful references

Software

Veriflow GitHub: <https://github.com/Deltares/veriflow>

Veriflow Documentation: <https://deltares.github.io/veriflow/>

Scores Documentation: <https://scores.readthedocs.io/en/stable/>

Papers

Murphy, Allan H. "What is a good forecast? An essay on the nature of goodness in weather forecasting." *Weather and forecasting* 8.2 (1993): 281-293.

Pagano, T.C., Ebert, E.E. and Khanarmuei, M. (2025), Enhancing Forecast Verification in National Meteorological and Hydrological Services. *Meteorol Appl*, 32: e70051. <https://doi.org/10.1002/met.70051>

Loveday, Nicholas, et al., 2024. "The Jive Verification System and its Transformative Impact on Weather Forecasting Operations". *Bull. Am. Met. Soc.* Sept. 11. <https://doi.org/10.1175/BAMS-D-23-0267.1>

Books

Wilks, Daniel S. *Statistical methods in the atmospheric sciences*. Vol. 100. Academic press, 2011.

TIGGE workshop ECMWF (Beth Ebert, BoM); <https://www.ecmwf.int/sites/default/files/elibrary/2005/15865-verification-ensembles.pdf>

Guidelines on the Verification of Hydrological Forecasts (WMO); <https://library.wmo.int/records/item/69478-guidelines-on-the-verification-of-hydrological-forecasts>

References

Veriflow (Deltares); <https://github.com/Deltares/veriflow>

TIGGE workshop ECMWF (Beth Ebert, BoM);

<https://www.ecmwf.int/sites/default/files/elibrary/2005/15865-verification-ensembles.pdf>

Guidelines on the Verification of Hydrological Forecasts (WMO);

<https://library.wmo.int/records/item/69478-guidelines-on-the-verification-of-hydrological-forecasts>



5 Discussion

Veriflow... and forecast verification in general

Would you want to do more on forecast verification?

Is there a role for veriflow there?

How do you see the role of the Delft-FEWS suite?

Stay in touch



Maarten Smoorenburg

Maarten.Smoorenburg@deltares.nl

<https://github.com/Deltares/veriflow>



www.deltares.nl



info@deltares.nl



[linkedin.com
/company
/deltares](https://www.linkedin.com/company/deltares)



[@deltares](https://www.instagram.com/deltares)



[facebook.com
/deltaresNL](https://www.facebook.com/deltaresNL)