Machine-Learning Improved Metocean Forecasting for All

Stephan Kistner[1] (stki@dhigroup.com), Jannik Elsäßer[1], Juame Palmer Real[1]
[1]DHI A/S, Hørsholm, Denmark

5th ECMWF-ESA Machine Learning Workshop, Bologna · April 2026

Problem

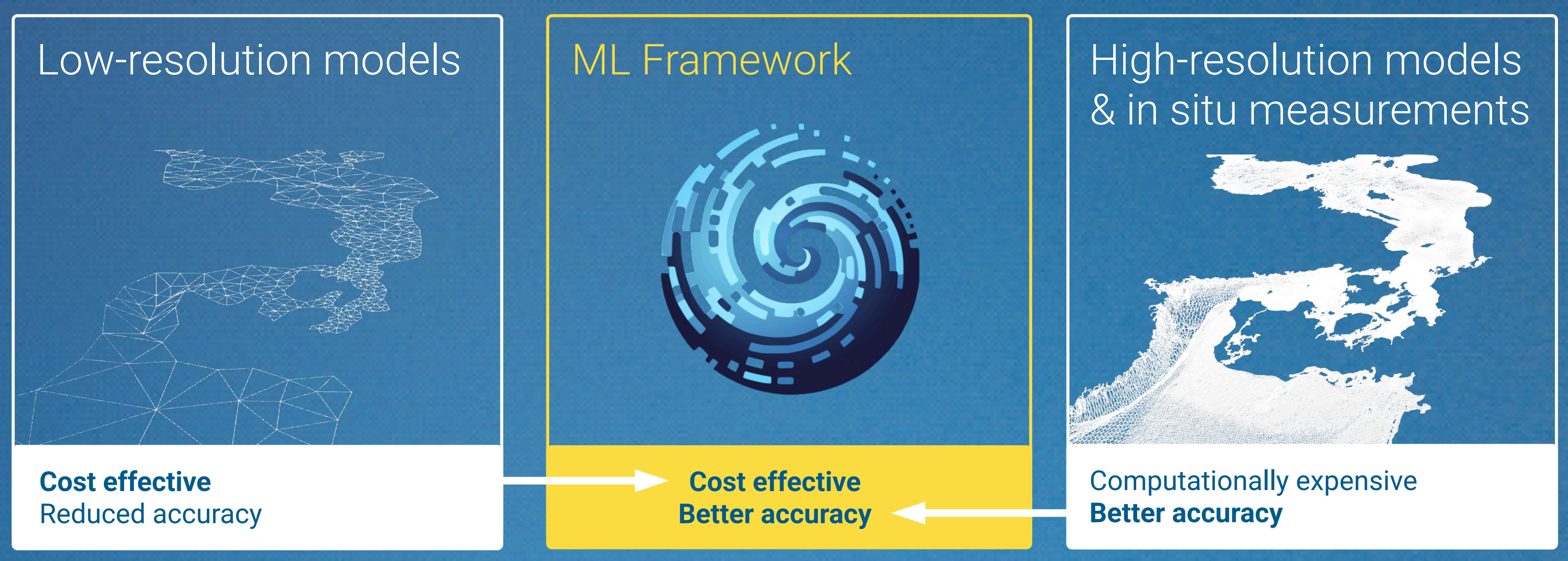
The cost of high-resolution models

Accurate metocean forecasts are increasingly important for offshore planning and operations

High-resolution local models are **costly and computationally demanding**

Freely available global models lack site-specific accuracy

This limits their use for operational decision support, particularly for SMEs



Low-resolution models
Cost effective
Reduced accuracy

ML Framework
Cost effective
Better accuracy

High-resolution models & in situ measurements
Computationally expensive
Better accuracy

Proposed solution

A pragmatic ML framework

A time-series-focused **user-friendly ML framework** for designed for metocean datasets

Improves freely available global forecasts via downscaling and non-linear bias correction

Designed for rapid operational use without specialist ML expertise or HPC

Supports deterministic and ensemble forecasts, using public or in-situ data

Approach

The ML framework architecture

Many ML libraries provide powerful functionality but present a high entry barrier for new users. The ML framework extracts those features, simplifies them, tailors them towards metocean forecasting, and makes them cross-compatible across libraries.

Modular Workflows

ML models and transforms are decoupled from the pipeline and stored as external files which can be hot-swapped in production without changes to the surrounding workflow.

Quick development

In many workflows, a substantial effort is required to organise and convert data. By handling these tasks internally, this framework significantly reduces setup time and enables faster development and testing.

Multiple libraries, one API

The ML framework extracts and standardises core model capabilities, exposing a simplified, consistent syntax with cross-compatible functionality for established libraries.

Designed for timeseries

The framework is explicitly scoped to timeseries modelling in the metocean domain.
Provides native support for temporal structure, including indexing, lags, leads and sliding windows.

Example workflow

A typical end-to-end workflow is as follows:

Prepare input data

Features and targets are saved in a structured Timeseries:

```
timeseries = Timeseries(
    features=[Hm0, Tp, MWD, WS, ],
    targets=[Hm0_measured])
```

The Timeseries structure is preserved throughout the workflow and supports interoperability across all included libraries.

Setup model

Model setup, training, and testing can be performed within a ModelFrame structure. The framework automatically handles the splitting of training, validation, and testing data, and ensures that pre- and post-processing steps are applied in the correct order.

```
model_frame = (
    ModelFrame("Demo ML Framework")
    .add_model(GBoost(…)
    .add_pre_transforms(…)
    .add_scaler(Scaler(…))
    .add_post_transforms(Function(…))
model_frame.fit_evaluate(timeseries)
```

```
ML architectures
```

`LinearModel()`
sklearn linear regression

`Dense()`
keras sequential dense neural network

`LSTM()`
keras sequential Long Short-Term Memory

`GBoost()`
sklearn gradient boost regression

`RandomForest()`
sklearn random forest regression

`...()`
flexible architecture with plans to add more

Operationalise

Saving the trained setup

The framework can be exported to a file.

```
model_frame.write(…)
```

Each of the individual framework elements are accessible and can also be exported.

```
model_frame.model.write(…)
model_frame.scaler.write(…)
model_frame.pre_transforms.write(…)
model_frame.post_transforms.write(…)
```

Production mode

Read the full model framework in one go.

```
model_frame = ModelFrame.read(…)
result = model_frame.predict(…)
```

Individual elements can be loaded independently, which enables operational features such as hot-swapping models in a production pipeline when improved setups become available.

```
model_frame.model = Model.read(…)
```

Key results

Hindcast data in the Southern North Sea

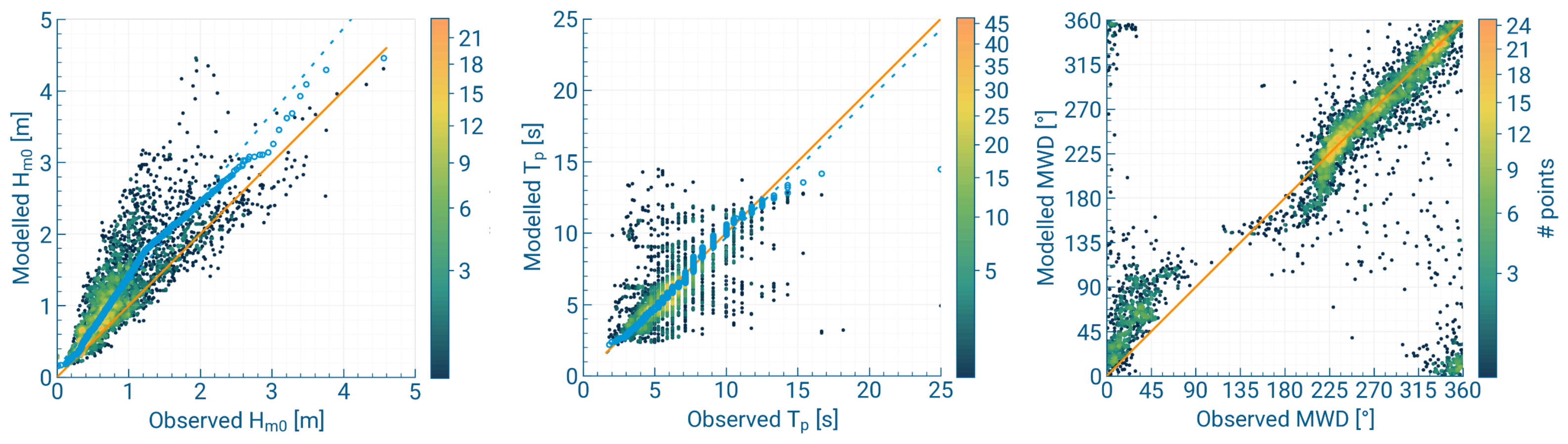
The WaterBench dataset [1] is a public benchmark comprising two years of wave model output and in-situ measurements. Observations were combined with ECMWF-ERA5 wave and atmospheric fields (0.25° resolution) [2] to evaluate ML framework performance.

The case study focuses on coastal locations where the ECMWF model shows reduced skill, using stations **6201059** (H_{m0}, T_p) and **WaddenEierlandseGat** (MWD) for validation.

Input features include ECMWF wave parameters (H_{m0}, T_p, T_{01}, T_{02}, MWD) and atmospheric variables (wind speed and direction); targets are measured H_{m0}, T_p, and MWD.

Comparing global model to ML-predictions

Raw global dataset



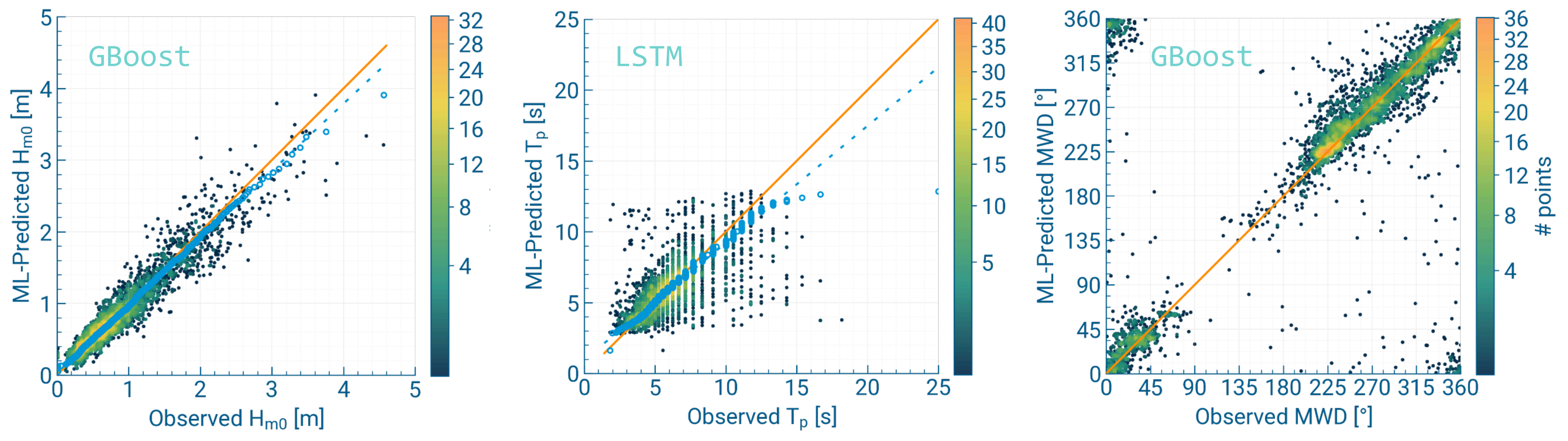
Metrics

Model	Bias	RMSE	MAE	CC	SI	R^2
Baseline (ERA5)	0.29	0.48	0.34	0.82	0.44	0.26
Dense	-0.02	0.17	0.11	0.96	0.19	0.91
GBoost	-0.02	0.15	0.11	0.96	0.17	0.93
LSTM	0.07	0.22	0.16	0.94	0.23	0.85
Random Forest	-0.03	0.15	0.11	0.96	0.17	0.92
Linear	-0.02	0.24	0.17	0.90	0.27	0.82

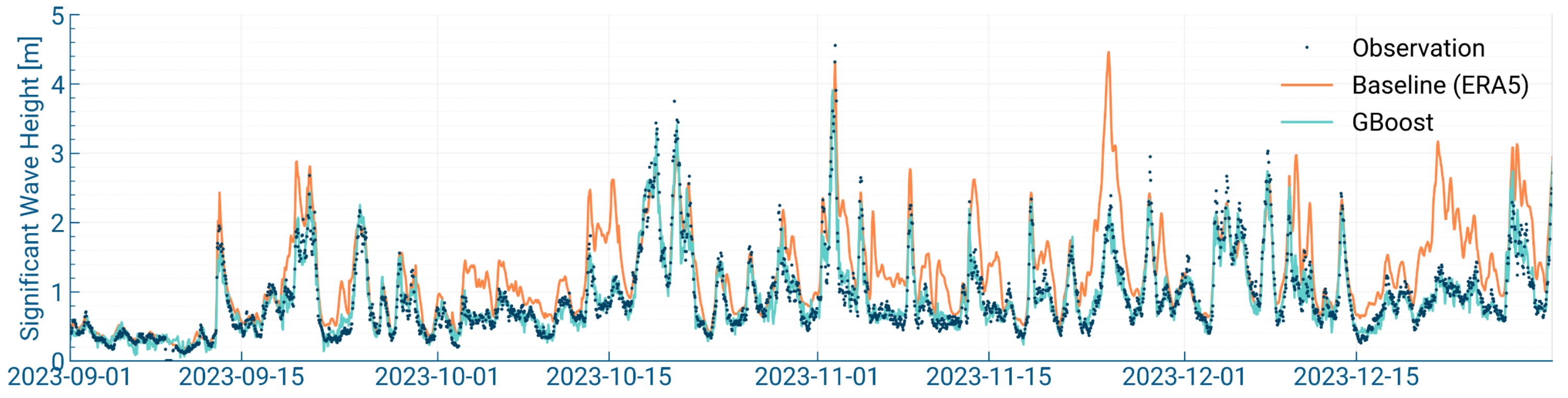
Model	Bias	RMSE	MAE	CC	SI	R^2
Baseline (ERA5)	-0.18	1.79	1.05	0.61	0.30	0.21
Dense	0.00	1.79	1.14	0.56	0.30	0.21
GBoost	-0.27	1.76	1.05	0.54	0.29	0.23
LSTM	-0.07	1.60	0.98	0.64	0.27	0.36
Random Forest	-0.29	1.58	0.95	0.64	0.26	0.39
Linear	-0.30	1.56	0.96	0.65	0.26	0.40

Model	Bias	RMSE	MAE
Baseline (ERA5)	3.4	29.8	18.6
Dense	2.3	21.1	12.7
GBoost	1.6	21.6	12.3
LSTM	1.6	22.1	13.1
Random Forest	1.5	22.6	12.8
Linear	0.7	22.6	13.5

ML-adjusted



Timeseries - raw global vs ML-corrected



For $\mathbf{H_{m0}}$, all machine-learning models improved performance relative to the baseline. GBoost showed the most substantial improvement across all performance metrics.

$\mathbf{T_p}$ proved more challenging to improve due to its discrete nature. The linear model achieved the best skill scores; however, the LSTM showed lower bias and a better distributional fit in comparison.

MWD showed marginal improvements; primarily confined to fringe directional sectors, particularly the north-easterly and southerly ranges.

Conclusions

ML-based time-series post-processing improves local wave estimates derived from global models

A time-series-native, modular framework enables quick setup, testing, and deployment for use in forecast correction and downscaling

Accuracy gains are achieved without specialised ML expertise or high-performance computing, and without replacing the underlying forecast model

The approach supports cost-effective decision support for offshore and renewable-energy applications

Code, examples and benchmark dataset

References

[1] DHI (2025). Spectral wave model of the Southern North Sea: MIKE 21 SW model setup, outputs, and observation data [Data set];https://doi.org/10.5281/zenodo.17413569

[2] Hersbach, H. et al. (2023). ERA5 hourly data on single levels from 1940 to present. Copernicus Climate Change Service (C3S) Climate Data Store; https://doi.org/10.24381/cds.adbb2d47