A Unified Benchmarking Platform for UQ Algorithms in UQLab

Other Conference Item

Author(s):
Hlobilová, Adéla; Marelli, Stefano; Sudret, Bruno

Publication date:
2024-02-29

Permanent link:
https://doi.org/10.3929/ethz-b-000662725

ETH zürich

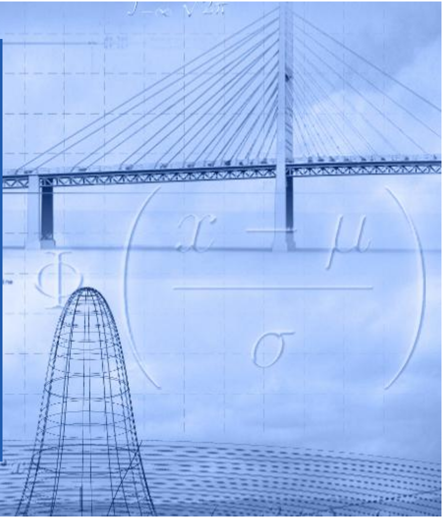
Risk, Safety & Uncertainty Quantification

A Unified Benchmarking Platform for UQ Algorithms in UQLab

SIAM UQ24

A. Hlobilová, S. Marelli & B. Sudret

February 29, 2024



Outline

1. Why is benchmarking essential?

2. Ingredients of a benchmark study

3. Demo No. 1: Live module presentation (A toy benchmark analysis)

4. Demo No. 2: Live module presentation (Post-processing of a real benchmark)

5. Summary

Why is benchmarking essential?

a novel surrogate model

About 749'000 results (**0.06** sec)

A novel sequential design strategy for global **surrogate modeling**
K Crombecq, L De Tommasi… - Proceedings of the …, 2009 - ieeexplore.ieee.org
… Please note that global **surrogate modeling** differs from local **surrogate modeling** in the way the **surrogate models** are employed. In local **surrogate modeling**, local **models** are used to …
Save Cite Cited by 156 Related articles All 11 versions

[HTML] Smart sampling algorithm for **surrogate model** development
SS Garud, IA Karimi, M Kraft - Computers & Chemical Engineering, 2017 - Elsevier
… Furthermore, numerical **models** … **model** into a computationally inexpensive **surrogate model** that captures its essential features with prescribed numerical accuracy. **Surrogate modelling**, …
Save Cite Cited by 82 Related articles All 3 versions Web of Science: 51

[HTML] **Surrogate modelling** for sustainable building design–A review
P Westermann, R Evins - Energy and Buildings, 2019 - Elsevier
… **surrogate model** type. Based on the literature major research trends are extracted and useful practical aspects outlined. As **surrogate modelling** … to make **surrogate modelling** accessible …
Save Cite Cited by 149 Related articles All 2 versions Web of Science: 81

[HTML] An adaptive hybrid **surrogate model**
J Zhang, S Chowdhury, A Messac - Structural and Multidisciplinary …, 2012 - Springer

a novel algorithm for reliability analysis

About 3'400'000 results (**0.21** sec)

A novel algorithm for structural **reliability analysis** based on finite step length and Armijo line search
P Huang, HZ Huang, T Huang - Applied Sciences, 2019 - mdpi.com
… This paper presents a **novel algorithm** for structural **reliability analysis** based on the finite … Rakwitz–Fiessler (HL-RF) **algorithm** that may be subjected to non-convergence in the first-order …
Save Cite Cited by 16 Related articles All 4 versions Web of Science: 11

[HTML] **Novel algorithms** for **reliability** evaluation of remotely deployed wireless sensor networks
C Chowdhury, N Aslam, G Ahmed… - Wireless Personal …, 2018 - Springer
… This paper investigates **reliability analysis** and makes two contributions. First, an **algorithm** based … We propose two **novel algorithms** to calculate **reliability**. An Ordered Binary Decision …
Save Cite Cited by 16 Related articles All 5 versions Web of Science: 8

A novel algorithm on network **reliability analysis**
J Xiong, W Gong - 10th International Conference on …, 2003 - ieeexplore.ieee.org
… a **novel algorithm** for network **reliability** analysis in this paper. The **algorithm** is based on rational approximation. It uses rational functions to estimate the transformed **reliability** function. It …
Save Cite Cited by 7 Related articles

Why is benchmarking essential?

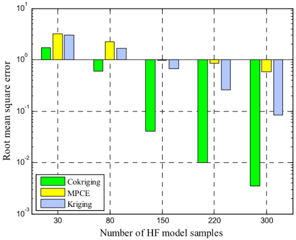


Fig. 1. RMSE of surrogate models for Ishigami function.

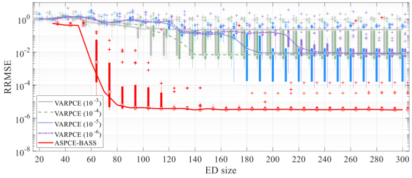


Fig. 4. RRMSE with increasing ED size.

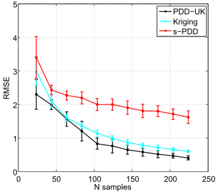


Fig. 3. TEST 1: Mean RMSE convergence comparison between Ordinary Kriging, sparse PDD and coupled PDD-UK metamodels, $m = 5$.

Shang, X. et al. (2023). An efficient multi-fidelity Kriging surrogate model-based method for global sensitivity analysis. Reliability Engineering & System Safety 229, 108858. doi

Zhang, B.-Y. & Ni, Y.-Q. (2023). A novel sparse polynomial chaos expansion technique with high adaptiveness for surrogate modelling. Applied Mathematical Modelling 121, 562-585. doi

Cortesi, A. F., Jannoun, G., & Congedo, P. M. (2019). Kriging-sparse Polynomial Dimensional Decomposition surrogate model with adaptive refinement. Journal of Computational Physics, 380, 212-242. doi

Why is benchmarking essential?

SIAM/ASA J. UNCERTAINTY QUANTIFICATION
Vol. 9, No. 2, pp. 593–649

Sparse Polynomial Chaos Expansions: Literature Survey and Benchmark*

Nora Lüthen[†], Stefano Marelli[†], and Bruno Sudret[†]

Abstract. Sparse polynomial chaos expansions (PCE) are a popular surrogate modelling method that takes advantage of the properties of PCE, the sparsity-of-effects principle, and powerful sparse regression solvers to approximate computer models with many input parameters, relying on only a few model evaluations. Within the last decade, a large number of algorithms for the computation of sparse PCE have been published in the applied math and engineering literature. We present an extensive review of the existing methods and develop a framework for classifying the algorithms. Furthermore, we conduct a unique benchmark on a selection of methods to identify which approaches work best in practical applications. Comparing their accuracy on several benchmark models of varying dimensionality and complexity, we find that the choice of sparse regression solver and sampling scheme for the computation of a sparse PCE surrogate can make a significant difference of up to several orders of magnitude in the resulting mean-squared error. Different methods seem to be superior in different regimes of model dimensionality and experimental design size.

Key words. uncertainty quantification, surrogate modelling, sparse regression, sparse polynomial chaos expan-

SIAM-ASA Journal on Uncertainty Quantification • *Open Access* • Volume 9, Issue 2, Pages 593 - 649 • 2021

Sparse polynomial chaos expansions: Literature survey and benchmark

Lüthen, Nora; Marelli, Stefano; Sudret, Bruno
Save all to author list

[a] Department of Risk, Safety and Uncertainty Quantification, ETH Zürich, Zürich, 8093, Switzerland

Structural Safety 96 (2022) 102174

Contents lists available at ScienceDirect
Structural Safety
journal homepage: www.elsevier.com/locate/strusafe

ELSEVIER

Active learning for structural reliability: Survey, general framework and benchmark

Maliki Moustapha *, Stefano Marelli, Bruno Sudret
Chair of Risk, Safety and Uncertainty Quantification, ETH Zurich, Stefano-Franscini-Platz 5, 8093 Zurich, Switzerland

ARTICLE INFO

Keywords:
Structural reliability
Active learning
Surrogate models
Benchmark

ABSTRACT

Active learning methods have recently surged in the literature due to their ability to solve complex structural reliability problems within an affordable computational cost. These methods are designed by adaptively building an inexpensive surrogate of the original limit-state function. Examples of such surrogates include Gaussian process models which have been adopted in many contributions, the most popular ones being the efficient global reliability analysis (EGRA) and the active Kriging Monte Carlo simulation (AK-MCS), two milestone

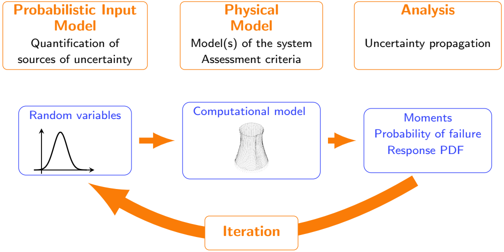
Structural Safety • *Open Access* • Volume 96 • May 2022 • Article number 102174

Active learning for structural reliability: Survey, general framework and benchmark

Moustapha, Maliki; Marelli, Stefano; Sudret, Bruno
Save all to author list

[a] Chair of Risk, Safety and Uncertainty Quantification, ETH Zurich, Stefano-Franscini-Platz 5, Zurich, 8093, Switzerland

The UQLab software framework



- Matlab-based high-level language
- Complete framework for uncertainty quantification
- BSD license: completely open-source for both academia and industry
- Approx. 6.8k unique users from 90+ countries
- 1500+ combined citations on Google Scholar since 2014

Continuously developed/updated by the RSUQ Chair @ETH Zurich

Marelli, S. & Sudret, B. (2014). UQLab: A framework for uncertainty quantification in Matlab. Proc. 2nd Int. Conf. on Vulnerability, Risk Analysis and Management (ICVRAM2014), Liverpool, United Kingdom, 2554-2563. doi:10.3929/ethz-a-010238238

Our proposal

- Unified, standardized, and objective way to evaluate the performance of algorithms against established standards
- Curated database containing datasets, configurations, algorithms, and performance measures
- Sets of settings of competitors tailored for specific benchmark case scenarios
- Reduce the workload and time investment for researchers

Outline

1. Why is benchmarking essential?

2. Ingredients of a benchmark study

3. Demo No. 1: Live module presentation (A toy benchmark analysis)

4. Demo No. 2: Live module presentation (Post-processing of a real benchmark)

5. Summary

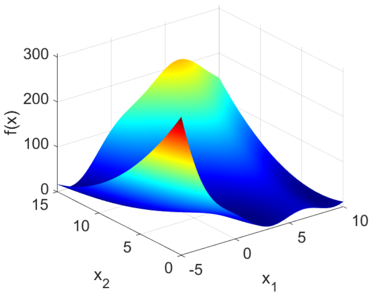
Ingredients of a benchmark study

- Bench cases
- Competitors
- Performance measures
- Post-processing of the results

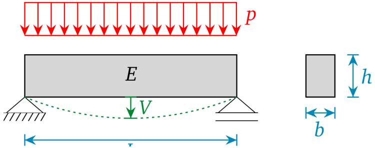
Additional features

- Library
- Dispatch of computations (HPC)

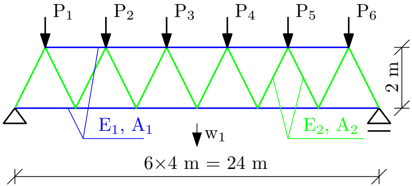
Bench cases



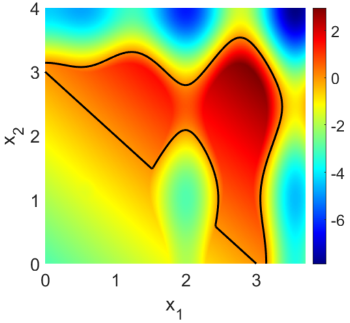
300
200
100
0
f(x)
15
10
5
0
x2
-5
0
5
10
x1



p
E
V
h
b



P1
P2
P3
P4
P5
P6
2 m
E1, A1
w1
E2, A2
6×4 m = 24 m



4
3
2
1
0
x2
0
1
2
3
x1
2
0
-2
-4
-6

Bench cases

- Described by
  - a computational model with an associated set of probability distributions, or
  - datasets
- Example: Borehole model for surrogate modeling

$$f(X) = \frac{2\pi T_u (H_u - H_l)}{\ln\left(\frac{r}{r_w}\right)\left(1 + \frac{2LT_u}{\ln\left(\frac{r}{r_w}\right) r_w^2 K_w} + \frac{T_u}{T_l}\right)}$$

Variable	Description	Distribution	Statistics
r_w	Radius of borehole (m)	Normal	μ_{rw} = 0.10, σ^2_{rw} = 0.0161812
r	Radius of influence (m)	Lognormal	μ_r = 7.71, σ_r = 1.0056
T_u	Transmissivity of upper aquifer (m^2/yr)	Uniform	$T_u \sim U(63070, 115600)$
H_u	Potentiometric head of upper aquifer (m)	Uniform	$H_u \sim U(990, 1110)$
T_l	Transmissivity of lower aquifer (m^2/yr)	Uniform	$T_l \sim U(63.1, 116)$
H_l	Potentiometric head of lower aquifer (m)	Uniform	$H_l \sim U(700, 820)$
L	Length of borehole (m)	Uniform	$L \sim U(1120, 1680)$
K_w	Hydraulic conductivity of borehole (m/yr)	Uniform	$K_w \sim U(9855, 12045)$

sample No.	r_w	r	T_u	H_u	T_l	H_l	L	K_w	$f(X)$
1	0.096852	3778.2	90956	1106.2	104.3	733.72	1410.1	11336	87.913
2	0.078719	3186	97448	1053.8	85.501	752.81	1648.4	10488	37.185
3	0.10404	354.06	84482	1029	67.905	763.26	1477.1	11326	68.951
4	0.09882	501.17	1.0614e5	1002.7	77.196	754.89	1656.3	11377	52.024
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
N	0.1083	1392.1	1.0081e5	1063.3	71.228	805.04	1254.8	10004	75.399

Competitors

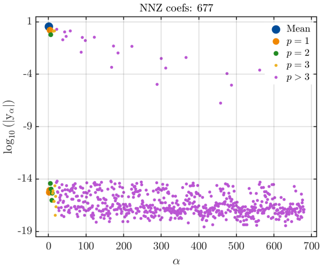
Potential competitors for benchmarking

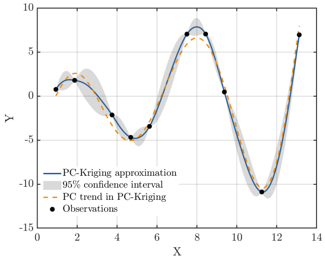
- Metamodels
- Reliability analysis
- Classifiers
- Samplers
- [Robust] Optimization algorithms
- Bayesian inference/inversion
- [Sensitivity analysis (specific measures, such as ANOVA)]

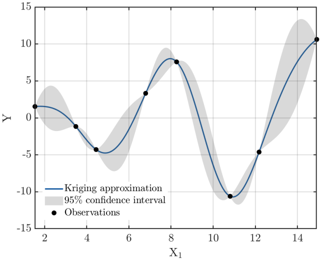
Competitors

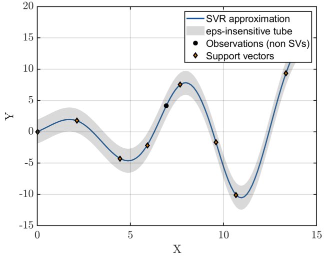
Metamodels

- Configuration parameters:
  - Metamodel family
  - Smoothness (degree/covariance)
  - Sparsity
  - Optimization algorithms
  - Adaptivity
- Example: Kriging
  - Correlation families (exponential/Matern/...)
  - Optimization algorithms (BFGS, GA)
  - Regression vs interpolation









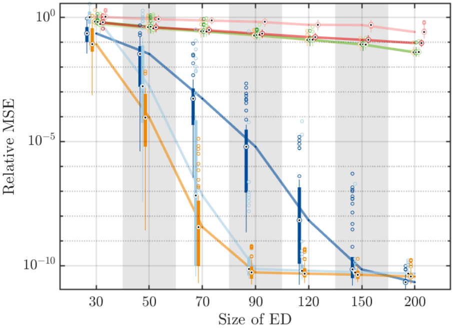
Performance measures

Why are performance measures computed?

- Comparing different algorithms
- Assessing accuracy and precision of the algorithm
- Improving algorithm development

Typical performance measures

- Accuracy prediction (e.g., RMSE, MAE, or Rˆ2)
- Convergence analysis (e.g., probability of failure or reliability index)
- Computational efficiency (costs)
- Robustness (e.g., by cross-validation)



Analysis, and aggregating & post-processing results

Let's run the benchmarking analysis!

- Assemble all combinations of bench cases and competitors
- Run the competitors to obtain performance measures

What should be included in post-processing?

- Brief overview of the analysis
- Aggregating results
- Book-keeping / Interactive results
- Ranking
- Graphical representation of results

Library

Purpose of the library

- To provide a **curated, standardized, and comprehensive collection** of benchmark cases, settings, and results for evaluating and comparing the performance of newly implemented algorithms

What should be stored in the library?

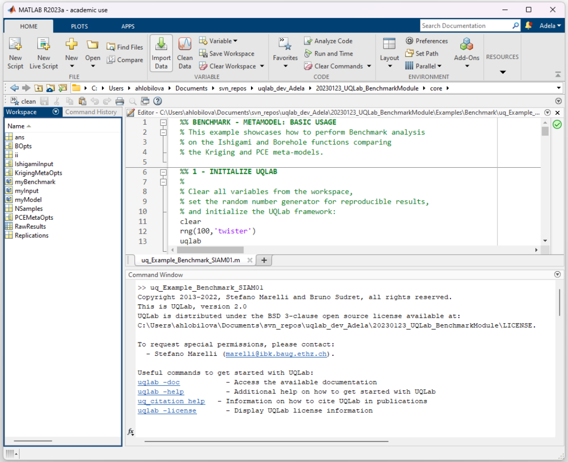
- **Bench case inputs:** Input data for benchmark cases
- **Competitor settings:** Settings used by previously analyzed competitors during benchmark analysis
- **Unified benchmark results:** Exhaustive collection of benchmark results
- **Curated competitor settings:** Specified sets of settings for competitors, tailored for specific benchmark cases

Outline

1. Why is benchmarking essential?
2. Ingredients of a benchmark study
3. Demo No. 1: Live module presentation (A toy benchmark analysis)
4. Demo No. 2: Live module presentation (Post-processing of a real benchmark)
5. Summary

Demo No. 1

Run the live demo in MATLAB



Demo No. 1
Initialization

Goal: Initialize UQLab and the Benchmark module using the Metamodel benchmark type

```
%% 1 - INITIALIZE UQLAB
%
% Clear all variables from the workspace,
% set the random number generator for reproducible results,
% and initialize the UQLab framework:
clear, clc
rng(100,'twister')
uqlab

%% 2 - SETUP THE SURROGATE MODELLING BENCHMARK
%
% Select the benchmark tool and the benchmark type case:
BOpts.Type = 'Benchmark';
BOpts.BenchmarkType = 'Metamodel';
```

Demo No. 1

Bench case: Ishigami function

Goal: Bench case #1 with 3 different experimental designs (30, 50, and 70 samples) and 10 different replications (by using a user-defined input and model)

```
%% 3 - BENCH CASES
%% 3.1 - ISHIGAMI FUNCTION
BOpts.BenchCases(1).Name = 'Ishigami function';
%% 3.1.1 - COMPUTATIONAL MODEL
BOpts.BenchCases(1).Model.mFile = 'uq_ishigami';

%% 3.1.2 - PROBABILISTIC INPUT MODEL
for ii = 1:3
    IshigamiInput.Marginals(ii).Type = 'Uniform';
    IshigamiInput.Marginals(ii).Parameters = [-pi pi];
end
BOpts.BenchCases(1).Input = IshigamiInput;

%% 3.1.3 - EXPERIMENTAL DESIGN AND VALIDATION SETS
NSamples = [30, 50, 70];
Replications = 10;
BOpts.BenchCases(1).ExpDesigns = struct('NSamples', num2cell(NSamples), ...
                                        'Replications', Replications);
BOpts.BenchCases(1).ValidationSet.NSamples = 1e4;
```

Demo No. 1

Bench case: Borehole function

Goal: Bench case #2 with 3 different experimental designs (30, 50, and 70 samples) and 10 different replications (by using UQLab-defined input and model)

```
%% 3.2 - BOREHOLE FUNCTION
uq_BoreholeInputModel;
BOpts.BenchCases(2).Name = 'Borehole function';

%% 3.2.1 - COMPUTATIONAL MODEL
BOpts.BenchCases(2).Model = myModel;

%% 3.2.2 - PROBABILISTIC INPUT MODEL
BOpts.BenchCases(2).Input = myInput;

%% 3.2.3 - EXPERIMENTAL DESIGN AND VALIDATION SETS
NSamples = [30, 50, 70];
Replications = 10;
BOpts.BenchCases(2).ExpDesigns = struct('NSamples', num2cell(NSamples), ...
                                        'Replications', Replications);
BOpts.BenchCases(2).ValidationSet.NSamples = 1e4;
```

Demo No. 1

Competitors: PCE, Kriging

Goal: Set up PCE and Kriging competitors with (almost) default options

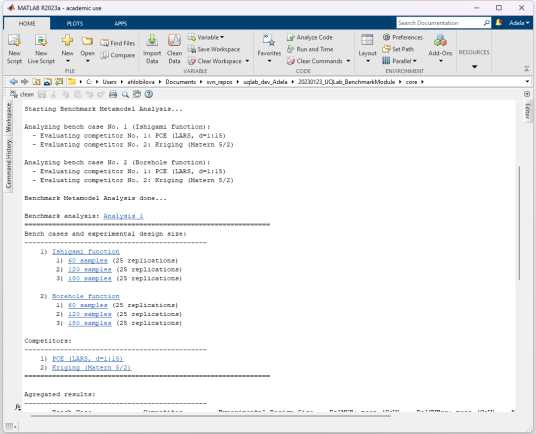
```
%% 4 - COMPETITORS

%% 4.1 PCE
PCEMetaOpts.Type = 'MetaModel';
PCEMetaOpts.MetaType = 'PCE';
PCEMetaOpts.Degree = 1:15;
PCEMetaOpts.Method= 'LARS';

BOpts.Competitors(1).MetaOpts = PCEMetaOpts;
BOpts.Competitors(1).Name = 'PCE (LARS, d=1:15)';

%% 4.2 Kriging
KrigingMetaOpts.Type = 'Metamodel';
KrigingMetaOpts.MetaType = 'Kriging';
KrigingMetaOpts.Corr.Family = 'Exponential';

BOpts.Competitors(2).MetaOpts = KrigingMetaOpts;
BOpts.Competitors(2).Name = 'Ordinary Kriging (Exponential corr. family)';
```

Demo No. 1

Run and post-process the analysis

```
%% 5 - GENERATE AND RUN THE BENCHMARK
myBenchmark = uq_createAnalysis(BOpts);

%% 6 - POSTPROCESSING
% Print a summary of the resulting analysis
uq_print(myBenchmark)

% Create a graphical representation of the results
uq_display(myBenchmark, 'full')

%% 7 - STORING BENCHMARKING ANALYSIS DATA
uq_saveToLibrary(myBenchmark,'myNewFancyLibrary.mat')
```

Demo No. 1
Results

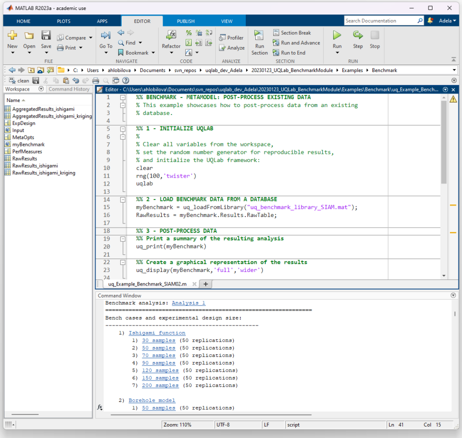


Outline

1. Why is benchmarking essential?
2. Ingredients of a benchmark study
3. Demo No. 1: Live module presentation (A toy benchmark analysis)
4. Demo No. 2: Live module presentation (Post-processing of a real benchmark)
5. Summary

Demo No. 2: Post-processing of a real benchmark

Run the live demo in MATLAB



Demo No. 2: Post-processing of a real benchmark
Bench cases

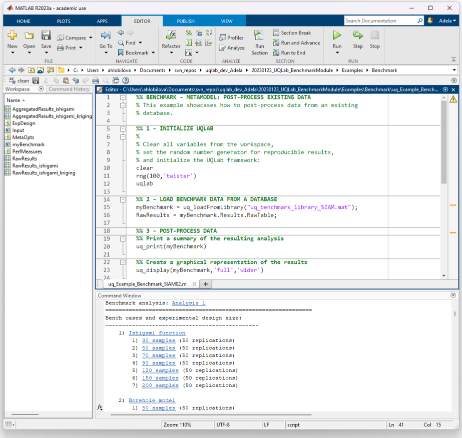
Bench case	**Dimension**	**Experimental designs (samples)**	**Validation set (samples)**
Ishigami function	3	30, 50, 70, 90, 120, 150, 200	100 000
Borehole function	8	50, 100, 150, 200, 250, 300	100 000
Damped oscillator	8	100, 150, 200, 250, 300, 350, 400	100 000
Wingweight function	10	100, 150, 200, 250, 300	100 000
Truss model	10	100, 150, 200, 250, 300	100 000
		50 replications	1 replication

Demo No. 2: Post-processing of a real benchmark
Competitors

No.	Competitor Names	Number of Hyperparameters
1	PCE (LARS)	3
2	PCE (OMP)	3
3	PCE (SP)	3
4	Ordinary Kriging	4
5	Kriging with a linear trend	4
6	PCK	7
7	XGBoost (sklearn Python module)	8
8	Multi-Layer Perceptron (torch Python module)	10

Demo No. 2: Post-processing of a real benchmark

Results in MATLAB



Outline

1. Why is benchmarking essential?
2. Ingredients of a benchmark study
3. Demo No. 1: Live module presentation (A toy benchmark analysis)
4. Demo No. 2: Live module presentation (Post-processing of a real benchmark)
5. Summary

Summary

We propose a flexible framework to fairly and efficiently compare the performance of surrogate models:

- Selection of benchmark cases
- Selection of competitors
- Natively enabling replications
- Standardized performance measures
- Integration of data from third-party packages
- Archiving through open-source libraries of pre-computed performances

Next steps

- Finalizing the current state (external competitors called by the benchmark module, asynchronous computation, etc.)
- Release the software into UQLab Version 2.2, scheduled for late Q2, 2024
- After populating the benchmark library, open access to contributors from different fields to cater to a larger audience
- Extend benchmarking functionality to different types of analysis (e.g., Bayesian calibration, sensitivity analysis, optimization, etc.)
- Post a public challenge for datasets and algorithms on our online community UQWorld, which counts hundreds of users from different fields of applied science and engineering

Questions?

Chair of Risk, Safety & Uncertainty Quantification
www.rsuq.ethz.ch

Thank you very much for your attention!

The Uncertainty Quantification Software
www.uqlab.com

UQLab

www.uqpylab.uq-cloud.io

UQCloud

The Uncertainty Quantification Community
www.uqworld.org

UQWorld

BACKUP SLIDES

Live demo No. 1: Results
Benchmark analysis info

```
>> uq_Example_Benchmark_SIAM01
Copyright 2013-2022, Stefano Marelli and Bruno Sudret, all rights reserved.
This is UQLab, version 2.0
UQLab is distributed under the BSD 3-clause open source license available at:
C:\Users\ahlobilova\Documents\svn_repos\uqlab_dev_Adela\20230123_UQLab_BenchmarkModule\LICENSE.

To request special permissions, please contact:
  - Stefano Marelli (marelli@ibk.baug.ethz.ch).

Useful commands to get started with UQLab:
uqlab -doc            - Access the available documentation
uqlab -help           - Additional help on how to get started with UQLab
uq_citation_help    - Information on how to cite UQLab in publications
uqlab -license        - Display UQLab license information


Initialization of the Benchmark Analysis done at 14-Feb-2024 15:59:58...

Starting Benchmark Metamodel Analysis...

Analyzing bench case No. 1 (Ishigami function):
  - Evaluating competitor No. 1: PCE (LARS, d=1:15)
  - Evaluating competitor No. 2: Kriging (Matern 5/2)

Analyzing bench case No. 2 (Borehole function):
  - Evaluating competitor No. 1: PCE (LARS, d=1:15)
  - Evaluating competitor No. 2: Kriging (Matern 5/2)

Benchmark Metamodel Analysis done in 5 minutes and 12 seconds
```

Live demo No. 1: Results

Benchmark analysis post-processing: `uq_print`

```
Benchmark analysis: Analysis 1
==============================================================
Bench cases and experimental design size:
----------------------------------------------
    1) Ishigami function
        1) 30 samples (10 replications)
        2) 50 samples (10 replications)
        3) 70 samples (10 replications)

    2) Borehole function
        1) 30 samples (10 replications)
        2) 50 samples (10 replications)
        3) 70 samples (10 replications)

Competitors:
----------------------------------------------
    1) PCE (LARS, d=1:15)
    2) Kriging (Matern 5/2)
==============================================================

Agregated results:
----------------------------------------------
```

Bench Case	Competitor	Experimental Design Size	RelMSE: mean (CoV)	RelCVErr: mean (CoV)	MAPE: mean (CoV)	MAE_norm: mean (CoV)
Ishigami function	PCE (LARS, d=1:15)	30	3.15e-01 (67%)	6.59e-02 (133%)	-1.38e+01 (-119%)	9.54e-02 (36%)
Ishigami function	PCE (LARS, d=1:15)	50	2.68e-02 (105%)	6.03e-03 (184%)	-2.56e+00 (-190%)	2.01e-02 (58%)
Ishigami function	PCE (LARS, d=1:15)	70	9.65e-04 (283%)	8.46e-05 (245%)	4.98e-02 (205%)	2.48e-03 (171%)
Ishigami function	Kriging (Matern 5/2)	30	8.61e-01 (95%)	3.98e-01 (38%)	-1.13e+01 (-299%)	1.69e-01 (40%)
Ishigami function	Kriging (Matern 5/2)	50	3.51e-01 (30%)	3.23e-01 (31%)	-1.14e+01 (-115%)	1.07e-01 (16%)
Ishigami function	Kriging (Matern 5/2)	70	2.38e-01 (37%)	2.24e-01 (33%)	-1.50e+01 (-73%)	8.57e-02 (13%)
Borehole function	PCE (LARS, d=1:15)	30	3.18e-02 (73%)	6.73e-03 (135%)	5.08e+00 (37%)	3.95e-03 (38%)
Borehole function	PCE (LARS, d=1:15)	50	2.27e-03 (31%)	1.11e-03 (49%)	1.37e+00 (11%)	1.02e-03 (17%)
Borehole function	PCE (LARS, d=1:15)	70	1.43e-03 (22%)	7.33e-04 (44%)	1.13e+00 (16%)	7.59e-04 (14%)
Borehole function	Kriging (Matern 5/2)	30	3.31e-02 (24%)	2.22e-02 (41%)	4.26e+00 (15%)	3.66e-03 (14%)
Borehole function	Kriging (Matern 5/2)	50	1.77e-02 (43%)	7.28e-03 (28%)	2.93e+00 (20%)	2.44e-03 (19%)
Borehole function	Kriging (Matern 5/2)	70	1.01e-02 (22%)	6.99e-03 (52%)	1.81e+00 (10%)	1.59e-03 (10%)

```
==============================================================

Useful functions for post-processing data in Benchmark module:
uq_printBenchmark     - Display interactive benchmark analysis
uq_filterBenchmark    - Filter benchmark data
uq_getDetails         - Get details about a particular test case
uq_aggregateResults   - Aggregate data from the benchmarking analysis for the replicated datasets
uq_saveToLibrary      - Save data to a new library or add data to an existing library
uq_loadFromLibrary    - Load data from a library to UQLab
==============================================================
```

Live demo No. 1: Results

Benchmark analysis post-processing: `uq_print`, filtering

Bench Case	Competitor	Experimental Design Size	RelMSE: mean (CoV)	RelCVErr: mean (CoV)	MAPE: mean (CoV)	MAE_norm: mean (CoV)
Ishigami function	PCE (LARS, d=1:15)	30	3.15e-01 (67%)	6.59e-02 (133%)	-1.38e+01 (-119%)	9.54e-02 (36%)
Ishigami function	PCE (LARS, d=1:15)	50	2.68e-02 (105%)	6.03e-03 (184%)	-2.56e+00 (-190%)	2.01e-02 (58%)
Ishigami function	PCE (LARS, d=1:15)	70	9.65e-04 (283%)	8.46e-05 (245%)	4.98e-02 (205%)	2.48e-03 (171%)
Ishigami function	Kriging (Matern 5/2)	30	8.61e-01 (95%)	3.98e-01 (38%)	-1.13e+01 (-299%)	1.69e-01 (40%)
Ishigami function	Kriging (Matern 5/2)	50	3.51e-01 (30%)	3.23e-01 (31%)	-1.14e+01 (-115%)	1.07e-01 (16%)
Ishigami function	Kriging (Matern 5/2)	70	2.38e-01 (37%)	2.24e-01 (33%)	-1.50e+01 (-73%)	8.57e-02 (13%)

```
Benchmark analysis: Analysis 1
==============================================================
Bench cases and experimental design size:
----------------------------------------------
    1) Ishigami function
        1) 30 samples (10 replications)
        2) 50 samples (10 replications)
        3) 70 samples (10 replications)

    2) Borehole function
        1) 30 samples (10 replications)
        2) 50 samples (10 replications)
        3) 70 samples (10 replications)

Competitors:
----------------------------------------------
    1) PCE (LARS, d=1:15)
    2) Kriging (Matern 5/2)
==============================================================
```

Live demo No. 1: Results

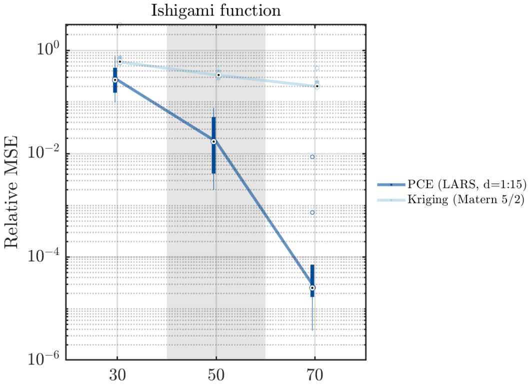
Benchmark analysis post-processing: `uq_print`, filtering

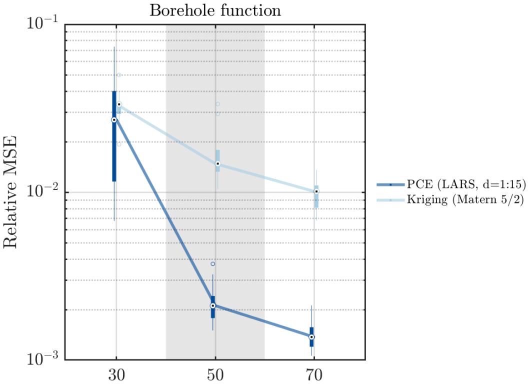
Bench Case	Competitor	Experimental Design Size	RelMSE: mean (CoV)	RelCVErr: mean (CoV)	MAPE: mean (CoV)	MAE_norm: mean (CoV)
Ishigami function	PCE (LARS, d=1:15)	30	3.15e-01 (67%)	6.59e-02 (133%)	-1.38e+01 (-119%)	9.54e-02 (36%)
Ishigami function	Kriging (Matern 5/2)	30	8.61e-01 (95%)	3.98e-01 (38%)	-1.13e+01 (-299%)	1.69e-01 (40%)

```
Benchmark analysis: Analysis 1
==============================================================
Bench cases and experimental design size:
----------------------------------------------
    1) Ishigami function
        1) 30 samples (10 replications)
        2) 50 samples (10 replications)
        3) 70 samples (10 replications)

    2) Borehole function
        1) 30 samples (10 replications)
        2) 50 samples (10 replications)
        3) 70 samples (10 replications)

Competitors:
----------------------------------------------
    1) PCE (LARS, d=1:15)
    2) Kriging (Matern 5/2)
==============================================================
```

Live demo No. 1: Results

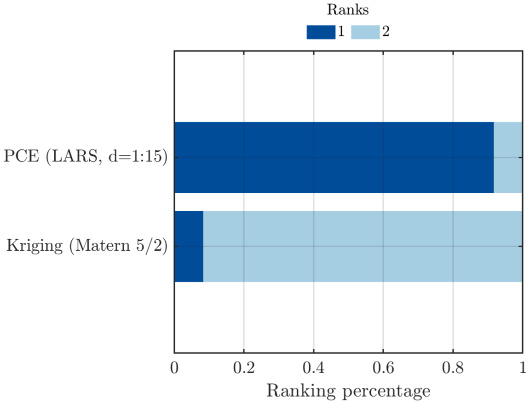
Benchmark analysis post-processing: `uq_display`

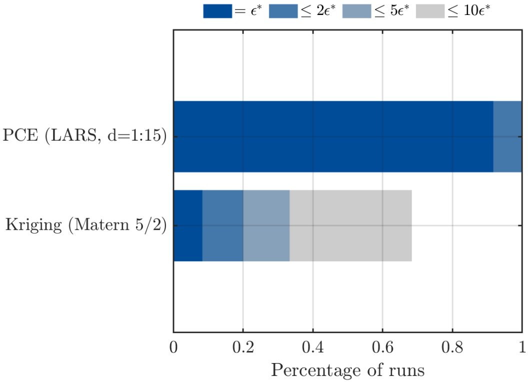




Live demo No. 1: Results

Benchmark analysis post-processing: `uq_display`





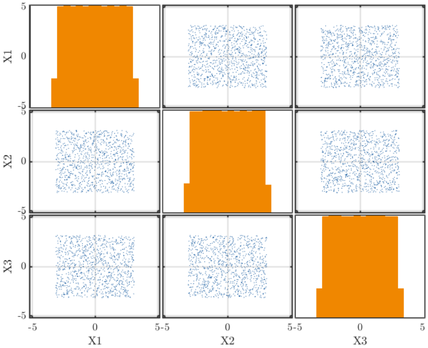
Live demo No. 2: Bench cases

Ishigami function B_1

The analytic expression:

$$Y(\mathbf{x}) = \sin(x_1) + 7\sin^2(x_2) + 0.1x_3^4\sin(x_1) \quad (1)$$

- 3 independent random variables $\mathbf{X} = \{X_1, X_2, X_3\}$
- Inputs are modelled by uniform distributions in the cube $\mathbf{X} \in [-\pi, \pi]^3$



	Number of Samples	Replications
Exp. Designs	30, 50, 70, 90, 120, 150, 200	50
Validation set	100 000	1

Live demo No. 2: Bench cases

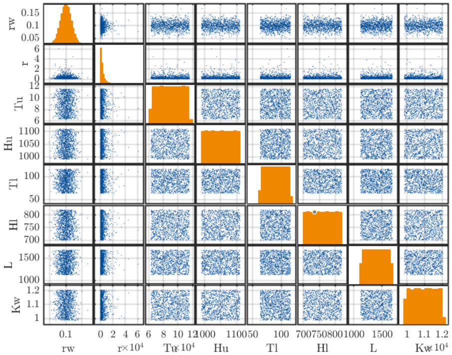
Borehole Function B_2

The analytic expression:

$$f(\mathbf{X}) = \frac{2\pi T_u(H_u - H_l)}{\ln(r/r_w)\left(1 + \frac{2LT_u}{\ln(r/r_w)r_w^2 K_w} + \frac{T_u}{T_l}\right)} \quad (2)$$

- 8 independent random variables $X = \{r_w, r, T_u, H_u, T_l, H_l, L, K_w\}^a$
- Inputs are modeled by normal (r_w), lognormal (r), and uniform distributions (T_u, H_u, T_l, H_l, L, K_w)

[a] where r_w: Radius of borehole, r: Radius of influence, T_u: Transmissivity of upper aquifer, H_u: Potentiometric head of upper aquifer, T_l: Transmissivity of lower aquifer, H_l: Potentiometric head of lower aquifer, L: Length of borehole, K_w: Hydraulic conductivity of borehole.

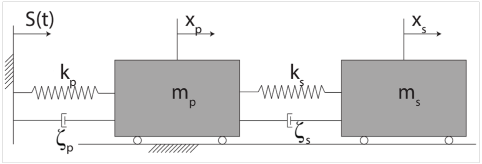


	Number of Samples	Replications
Exp. Designs	50, 100, 150, 200, 250, 300	50
Validation set	100 000	1

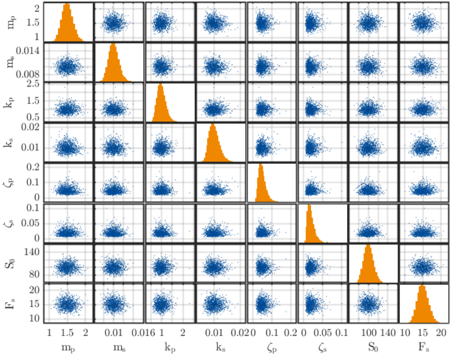
Live demo No. 2: Bench cases

Damped Oscillator Function B_3

Model:



- 8 independent lognormal distributed random variables
  $X = \{m_p, m_s, k_p, k_s, \zeta_p, \zeta_s, S_0, F_s\}^a$



	Number of Samples	Replications
Exp. Designs	100, 150, 200, 250, 300, 350, 400	50
Validation set	100 000	1

---

[a] where m_p: Primary mass, m_s: Secondary mass, k_p: Primary spring stiffness, k_s: Secondary spring stiffness, ζ_p: Primary damping ratio, ζ_s: Secondary damping ratio, S_0: Intensity of the white noise, F_s: Force capacity of the secondary spring.

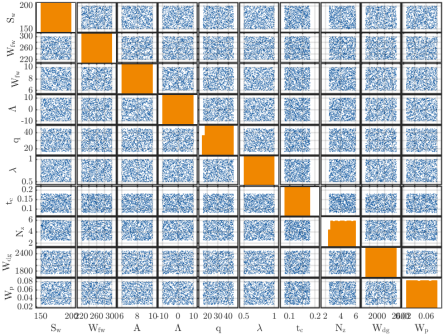
Live demo No. 2: Bench cases

Wing weight function B_4

The analytic expression:

$$f(\mathbf{X}) = 0.035 S_w^{0.758} W_{fw}^{0.0035} \left(\frac{A}{\cos^2(\Lambda)} \right)^{0.6} q^{0.006} \times \lambda^{0.04} \left(\frac{100 t_c}{\cos(\Lambda)} \right)^{-0.3} (N_z W_{dg})^{0.49} + S_w W_p \quad (3)$$

- 10 independent uniform distributed random variables $\mathbf{X} = \{S_w, W_{fw}, A, \Lambda, q, \lambda, t_c, N_z, W_{dg}, W_p\}^a$



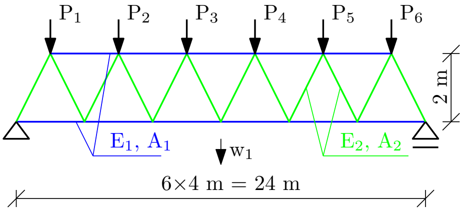
	Number of Samples	Replications
Exp. Designs	100, 150, 200, 250, 300	50
Validation set	100 000	1

[a] where S_w: Wing area (ft^2), W_{fw}: Weight of fuel in the wing (lb), A: Aspect ratio, Λ: Quarter-chord sweep (degrees), q: Dynamic pressure at cruise (lb/ft^2), λ: Taper ratio, t_c: Aerofoil thickness to chord ratio, N_z: Ultimate load factor, W_{dg}: Flight design gross weight (lb), W_p: Paint weight (lb/ft^2).

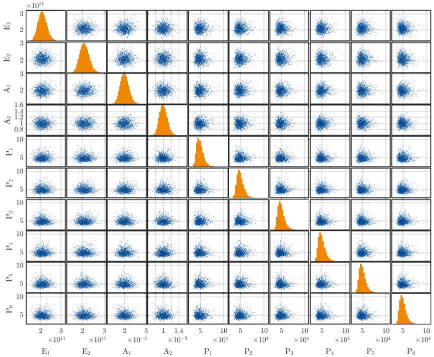
Live demo No. 2: Bench cases

Truss model B_5

Model:



- 10 independent random variables
  $X = \{E_1, E_2, A_1, A_2, P_1, P_2, P_3, P_4, P_5, P_6\}$[a]
- Inputs are modeled by lognormal (E_1, E_2, A_1, A_2) and Gumbel distributions $(P_i, i = 1, \ldots, 6)$



	Number of Samples	Replications
Exp. Designs	100, 150, 200, 250, 300	50
Validation set	100 000	1

[a] where A_1, A_2: Cross-sectional areas, E_1, E_2: Young's moduli, $P_i, i = 1, \ldots, 6$: applied loads.

Live demo No. 2: Competitors

UQLab

PCE

- Regression solvers: LARS C_1, OMP C_2, and SP C_3
- Degree adaptivity: $p \in [1, 15]$ with early stop based on ϵ_{LOO}
- q-norm truncation: $[0, 75, 1]$

Kriging

- ordinary C_4 and with linear trend C_5
- Correlation family: Matérn-5/2
- Correlation type: Ellipsoidal
- HGA optimizer

Sequential PC-Kriging C_6

- Kriging
  - ordinary Kriging
  - Correlation family: Matérn-5/2
  - Correlation type: Ellipsoidal
  - HGA optimizer
- PCE
  - Regression solver: LARS
  - Degree: 3

Live demo No. 2: Competitors
External

XGBoost C_7

- sklearn Python module
- `gamma`: $3.4 \cdot 10^{-7}$
- `learning_rate`: 0.13
- `max_depth`: 3
- `min_child_weight`: 1
- `n_estimators`: 900
- `reg_alpha`: $5.7 \cdot 10^{-4}$
- `reg_lambda`: 1.2
- `subsample`: 0.8

Multi-Layer Perceptron (MLP) C_8

- torch Python module
- `lr`: $5 \cdot 10^{-4}$
- `max_epochs`: 100
- `batch_size`: 256
- `optimizer_name`: `adamw`
- `lr_scheduler`: `False`
- `es_patience`: 40
- `lr_patience`: 30
- `module__n_layers`: 4
- `module__d_layers`: 1024
- `module__dropout`: 0.2

Live demo results: Example 2

Benchmark analysis post-processing: `uq_print`

```
Benchmark analysis: Analysis 1
==============================================================
Bench cases and experimental design size:
----------------------------------------------
    1) Ishigami function
        1) 30 samples (50 replications)
        2) 50 samples (50 replications)
        3) 70 samples (50 replications)
        4) 90 samples (50 replications)
        5) 120 samples (50 replications)
        6) 150 samples (50 replications)
        7) 200 samples (50 replications)

    2) Borehole model
        1) 50 samples (50 replications)
        2) 100 samples (50 replications)
        3) 150 samples (50 replications)
        4) 200 samples (50 replications)
        5) 250 samples (50 replications)
        6) 300 samples (50 replications)

    3) Damped oscillator
        1) 100 samples (50 replications)
        2) 150 samples (50 replications)
        3) 200 samples (50 replications)
        4) 250 samples (50 replications)
        5) 300 samples (50 replications)
        6) 350 samples (50 replications)
        7) 400 samples (50 replications)

    4) Wingweight function
        1) 100 samples (50 replications)
        2) 150 samples (50 replications)
        3) 200 samples (50 replications)
        4) 250 samples (50 replications)
        5) 300 samples (50 replications)

    5) Truss model
        1) 100 samples (50 replications)
        2) 150 samples (50 replications)
        3) 200 samples (50 replications)
        4) 250 samples (50 replications)
        5) 300 samples (50 replications)

Competitors:
----------------------------------------------
    1) PCE (LARS)
    2) PCE (OMP)
    3) PCE (SP)
    4) Ordinary Kriging
    5) Kriging with linear trend
    6) PCK
    7) XGB (ext., sklearn)
    8) MLP (ext., torch)
==============================================================
```

Live demo No. 2: Results

Benchmark analysis post-processing: `uq_print` (240 aggregated combinations)

Agregated results:

Bench Case	Competitor	Experimental Design Size	RelMSE: mean (CoV)	RelCVErr: mean (CoV)	MAPE: mean (CoV)	MAE_norm: mean (CoV)
Ishigami function	PCE (LARS)	30	2.80e-01 (77%)	4.45e-02 (117%)	2.23e+01 (62%)	8.88e-02 (43%)
Ishigami function	PCE (LARS)	50	5.40e-02 (151%)	7.45e-03 (171%)	8.52e+00 (91%)	2.83e-02 (82%)
Ishigami function	PCE (LARS)	70	3.94e-03 (345%)	4.69e-04 (314%)	1.60e+00 (106%)	5.27e-03 (146%)
Ishigami function	PCE (LARS)	90	1.41e-04 (279%)	2.73e-05 (239%)	3.33e-01 (174%)	8.95e-04 (156%)
Ishigami function	PCE (LARS)	120	9.50e-07 (313%)	2.98e-07 (335%)	2.00e-02 (236%)	6.24e-05 (223%)
Ishigami function	PCE (LARS)	150	2.14e-08 (354%)	2.42e-09 (247%)	3.06e-03 (206%)	8.02e-06 (186%)
Ishigami function	PCE (LARS)	200	2.60e-11 (61%)	1.07e-11 (47%)	3.25e-04 (45%)	8.64e-07 (11%)
Ishigami function	PCE (OMP)	30	1.19e+00 (70%)	3.86e-04 (363%)	5.77e+01 (61%)	1.86e-01 (45%)
Ishigami function	PCE (OMP)	50	1.61e-01 (239%)	2.85e-05 (529%)	1.15e+01 (199%)	3.37e-02 (179%)
Ishigami function	PCE (OMP)	70	1.09e-02 (453%)	8.97e-07 (480%)	1.12e+00 (280%)	3.35e-03 (281%)
Ishigami function	PCE (OMP)	90	5.45e-03 (707%)	1.32e-10 (477%)	3.41e-01 (708%)	6.93e-04 (692%)
Ishigami function	PCE (OMP)	120	1.69e-06 (568%)	1.01e-08 (706%)	8.08e-03 (572%)	2.18e-05 (485%)
Ishigami function	PCE (OMP)	150	4.02e-09 (696%)	3.36e-13 (136%)	-5.20e-05 (-6168%)	1.74e-06 (197%)
Ishigami function	PCE (OMP)	200	5.49e-11 (47%)	7.01e-13 (91%)	4.32e-04 (37%)	1.17e-06 (12%)
Ishigami function	PCE (SP)	30	1.98e-01 (121%)	1.99e-02 (219%)	1.95e+01 (73%)	6.19e-02 (60%)
Ishigami function	PCE (SP)	50	5.95e-04 (205%)	8.82e-05 (189%)	7.75e-01 (121%)	2.26e-03 (99%)
Ishigami function	PCE (SP)	70	6.09e-07 (353%)	8.27e-08 (380%)	1.78e-02 (282%)	3.96e-05 (246%)
Ishigami function	PCE (SP)	90	1.15e-10 (138%)	3.51e-11 (421%)	3.97e-04 (53%)	1.21e-06 (28%)
Ishigami function	PCE (SP)	120	8.62e-11 (124%)	7.57e-12 (212%)	4.10e-04 (47%)	1.16e-06 (20%)
Ishigami function	PCE (SP)	150	5.00e-11 (58%)	4.12e-12 (52%)	3.72e-04 (42%)	1.09e-06 (13%)
Ishigami function	PCE (SP)	200	4.73e-11 (72%)	3.97e-12 (38%)	3.43e-04 (46%)	1.02e-06 (14%)
Ishigami function	Ordinary Kriging	30	7.60e-01 (38%)	4.24e-01 (44%)	4.78e+01 (47%)	1.65e-01 (19%)
Ishigami function	Ordinary Kriging	50	4.46e-01 (46%)	2.90e-01 (43%)	4.37e+01 (58%)	1.18e-01 (19%)
Ishigami function	Ordinary Kriging	70	2.88e-01 (37%)	1.94e-01 (32%)	3.21e+01 (50%)	9.33e-02 (15%)
Ishigami function	Ordinary Kriging	90	1.90e-01 (23%)	1.41e-01 (30%)	2.47e+01 (39%)	7.55e-02 (12%)
Ishigami function	Ordinary Kriging	120	1.29e-01 (29%)	1.02e-01 (32%)	1.84e+01 (51%)	5.88e-02 (16%)
Ishigami function	Ordinary Kriging	150	8.15e-02 (23%)	6.86e-02 (31%)	1.37e+01 (47%)	4.42e-02 (13%)
Ishigami function	Ordinary Kriging	200	4.27e-02 (31%)	3.70e-02 (29%)	1.04e+01 (46%)	2.98e-02 (10%)
Ishigami function	Kriging with linear trend	30	7.35e-01 (30%)	4.39e-01 (43%)	5.74e+01 (45%)	1.64e-01 (17%)
Ishigami function	Kriging with linear trend	50	4.73e-01 (47%)	3.06e-01 (45%)	4.68e+01 (57%)	1.21e-01 (22%)
Ishigami function	Kriging with linear trend	70	3.01e-01 (36%)	2.01e-01 (33%)	3.37e+01 (47%)	9.52e-02 (16%)
Ishigami function	Kriging with linear trend	90	1.95e-01 (24%)	1.43e-01 (29%)	2.52e+01 (39%)	7.63e-02 (12%)
Ishigami function	Kriging with linear trend	120	1.33e-01 (29%)	1.04e-01 (33%)	1.83e+01 (51%)	5.95e-02 (16%)
Ishigami function	Kriging with linear trend	150	8.06e-02 (21%)	6.94e-02 (32%)	1.35e+01 (48%)	4.36e-02 (12%)
Ishigami function	Kriging with linear trend	200	4.26e-02 (31%)	3.71e-02 (29%)	1.04e+01 (46%)	2.97e-02 (10%)
Ishigami function	PCK	30	6.14e-01 (43%)	2.68e-01 (56%)	4.47e+01 (50%)	1.48e-01 (23%)
Ishigami function	PCK	50	4.11e-01 (58%)	1.90e-01 (54%)	3.89e+01 (62%)	1.14e-01 (23%)
Ishigami function	PCK	70	2.83e-01 (45%)	1.61e-01 (33%)	2.77e+01 (52%)	9.38e-02 (18%)
Ishigami function	PCK	90	2.11e-01 (32%)	1.33e-01 (24%)	2.28e+01 (46%)	7.81e-02 (14%)
Ishigami function	PCK	120	1.36e-01 (40%)	1.04e-01 (29%)	1.78e+01 (58%)	5.94e-02 (16%)
Ishigami function	PCK	150	8.22e-02 (22%)	6.74e-02 (26%)	1.40e+01 (47%)	4.38e-02 (11%)
Ishigami function	PCK	200	4.31e-02 (33%)	3.76e-02 (30%)	1.04e+01 (45%)	2.95e-02 (9%)
Ishigami function	XGB (ext., sklearn)	30	6.46e-01 (20%)	8.77e-01 (26%)	4.81e+01 (42%)	1.63e-01 (10%)
Ishigami function	XGB (ext., sklearn)	50	4.24e-01 (24%)	5.44e-01 (26%)	3.99e+01 (39%)	1.26e-01 (11%)
Ishigami function	XGB (ext., sklearn)	70	3.18e-01 (20%)	3.81e-01 (25%)	3.31e+01 (37%)	1.06e-01 (9%)
Ishigami function	XGB (ext., sklearn)	90	2.22e-01 (18%)	3.16e-01 (25%)	2.66e+01 (30%)	8.75e-02 (9%)

Live demo No. 2: Results

Benchmark analysis post-processing: `uq_filterBenchmark`

```
RawResults_ishigami = uq_filterBenchmark(RawResults, 'Ishigami function');
AggregatedResults_ishigami = uq_aggregateResults(RawResults_ishigami,'mean')
```

AggregatedResults_ishigami =

56×11 table

BenchCaseID	CompetitorID	ExpDesignID	mean_RMSE_norm	mean_MSE_norm	mean_NRMSE	mean_R2	mean_MAE_norm	mean_MAPE	mean_RelMSE	mean_RelCVErr
1	1	1	0.49167	0.28046	0.064675	0.71954	0.088833	22.301	0.28046	0.044531
1	1	2	0.17981	0.05399	0.023653	0.94601	0.028269	8.5187	0.05399	0.0074459
1	1	3	0.03547	0.0039448	0.0046658	0.99606	0.0052681	1.5968	0.0039448	0.00046906
1	1	4	0.0061094	0.00014146	0.00080364	0.99986	0.00089528	0.33343	0.00014146	2.7346e-05
1	1	5	0.00042684	9.4951e-07	5.6148e-05	1	6.2414e-05	0.020025	9.4951e-07	2.9835e-07
1	1	6	6.3117e-05	2.14e-08	8.3026e-06	1	8.0172e-06	0.0030562	2.14e-08	2.4248e-09
1	1	7	4.9407e-06	2.601e-11	6.4992e-07	1	8.6422e-07	0.0003253	2.601e-11	1.0692e-11
1	2	1	1.0105	1.1893	0.13293	-0.18925	0.18605	57.659	1.1893	0.00038583
1	2	2	0.21353	0.16141	0.028089	0.83859	0.033721	11.486	0.16141	2.8537e-05
1	2	3	0.03384	0.010917	0.0044514	0.98908	0.0033517	1.117	0.010917	8.975e-07
1	2	4	0.010645	0.0054506	0.0014003	0.99455	0.00069278	0.34119	0.0054506	1.3165e-10
1	2	5	0.0002579	1.6913e-06	3.3926e-05	1	2.1807e-05	0.0080849	1.6913e-06	1.0135e-08
:	:	:	:	:	:	:	:	:	:	:
1	7	3	0.56137	0.31835	0.073844	0.68165	0.10563	33.136	0.31835	0.38106
1	7	4	0.4687	0.2215	0.061654	0.7785	0.087496	26.591	0.2215	0.31628
1	7	5	0.39858	0.16025	0.05243	0.83975	0.073017	22.09	0.16025	0.22256
1	7	6	0.35996	0.13074	0.04735	0.86926	0.064542	17.685	0.13074	0.16989
1	7	7	0.30449	0.093363	0.040053	0.90664	0.053292	16.547	0.093363	0.13194
1	8	1	1.0562	1.1324	0.13893	-0.13237	0.22194	76.59	1.1324	1.5397
1	8	2	0.93773	0.88578	0.12335	0.11422	0.19704	65.929	0.88578	1.0325
1	8	3	0.87094	0.7637	0.11457	0.2363	0.18398	66.722	0.7637	0.83987
1	8	4	0.80413	0.65369	0.10578	0.34631	0.16886	55.15	0.65369	0.76533
1	8	5	0.72043	0.52592	0.094767	0.47408	0.15111	42.27	0.52592	0.603
1	8	6	0.69829	0.50004	0.091855	0.49996	0.14944	43.939	0.50004	0.51364
1	8	7	0.55491	0.32314	0.072994	0.67686	0.11602	36.606	0.32314	0.43724

Display all 56 rows.

Live demo No. 2: Results

Benchmark analysis post-processing: `uq_filterBenchmark`

```
RawResults_ishigami_kriging  = uq_filterBenchmark(RawResults, ...
    {'Ishigami function','kriging'},'-intersection');
AggregatedResults_ishigami_kriging = uq_aggregateResults(...
    RawResults_ishigami_kriging,'mean')
```

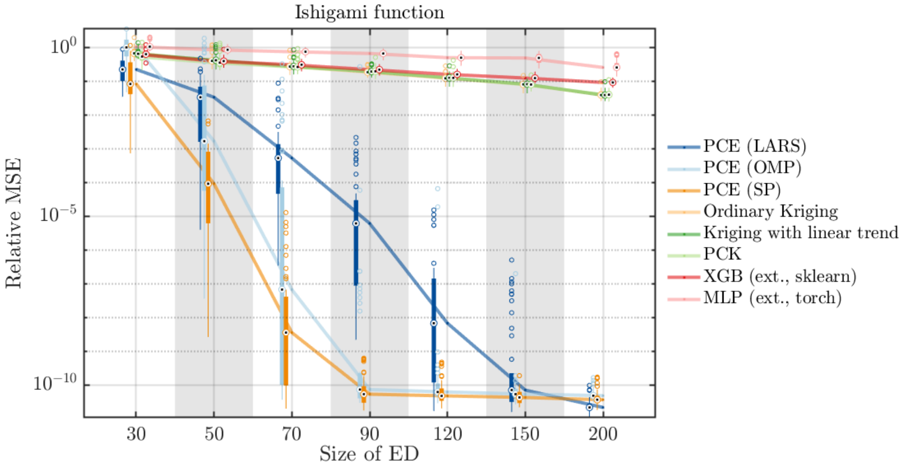
AggregatedResults_ishigami_kriging =

14×11 table

BenchCaseID	CompetitorID	ExpDesignID	mean_RMSE_norm	mean_MSE_norm	mean_NRMSE	mean_R2	mean_MAE_norm	mean_MAPE	mean_RelMSE	mean_RelCVErr
1	4	1	0.85876	0.76026	0.11296	0.23974	0.1647	47.82	0.76026	0.42431
1	4	2	0.65517	0.44611	0.086183	0.55389	0.11797	43.73	0.44611	0.29012
1	4	3	0.53001	0.28814	0.06972	0.71186	0.093297	32.088	0.28814	0.19373
1	4	4	0.4327	0.18962	0.056919	0.81038	0.075499	24.652	0.18962	0.14088
1	4	5	0.35572	0.12895	0.046793	0.87105	0.058793	18.416	0.12895	0.10198
1	4	6	0.28371	0.081513	0.03732	0.91849	0.044159	13.689	0.081513	0.068565
1	4	7	0.20455	0.042719	0.026907	0.95728	0.029821	10.398	0.042719	0.037018
1	5	1	0.84821	0.7348	0.11158	0.2652	0.16436	57.448	0.7348	0.43932
1	5	2	0.67301	0.47329	0.08853	0.52671	0.1214	46.762	0.47329	0.30595
1	5	3	0.54216	0.30141	0.071318	0.69859	0.095244	33.72	0.30141	0.201
1	5	4	0.43876	0.19514	0.057716	0.80486	0.076345	25.184	0.19514	0.14334
1	5	5	0.36101	0.13287	0.047488	0.86713	0.059472	18.272	0.13287	0.10411
1	5	6	0.28236	0.080606	0.037143	0.91939	0.04363	13.469	0.080606	0.069371
1	5	7	0.20426	0.042604	0.026869	0.9574	0.0297	10.428	0.042604	0.037112

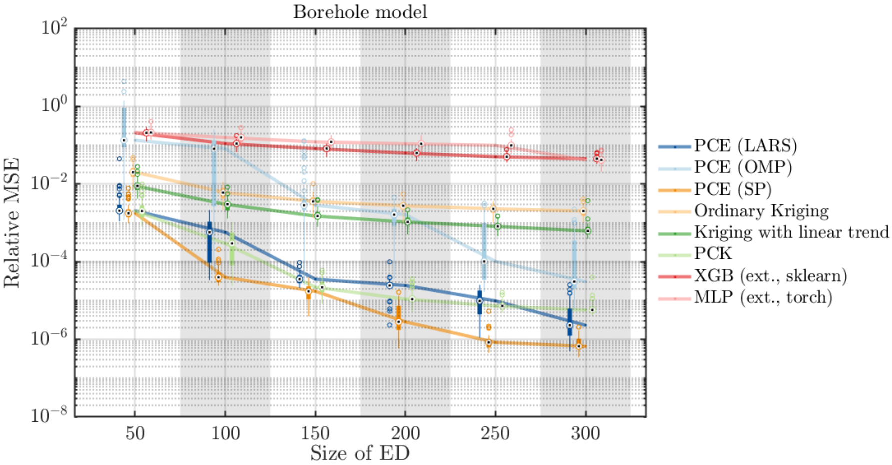
Live demo No. 2: Results

Benchmark analysis post-processing: `uq_display`



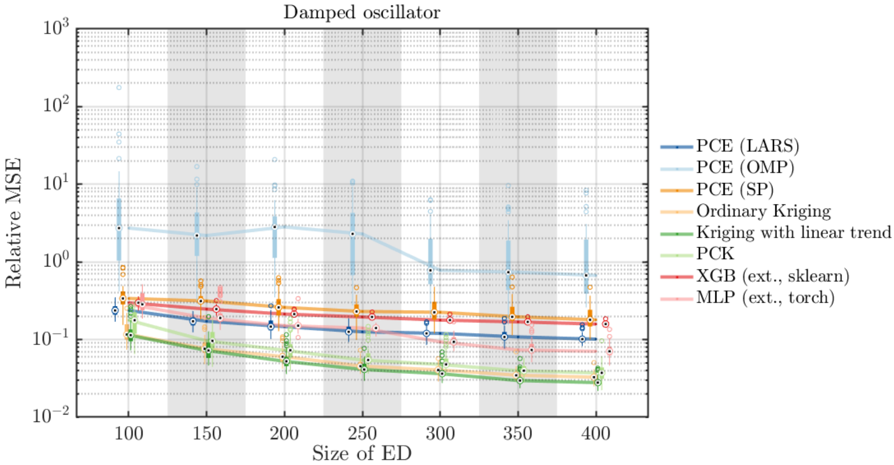
Live demo No. 2: Results

Benchmark analysis post-processing: `uq_display`



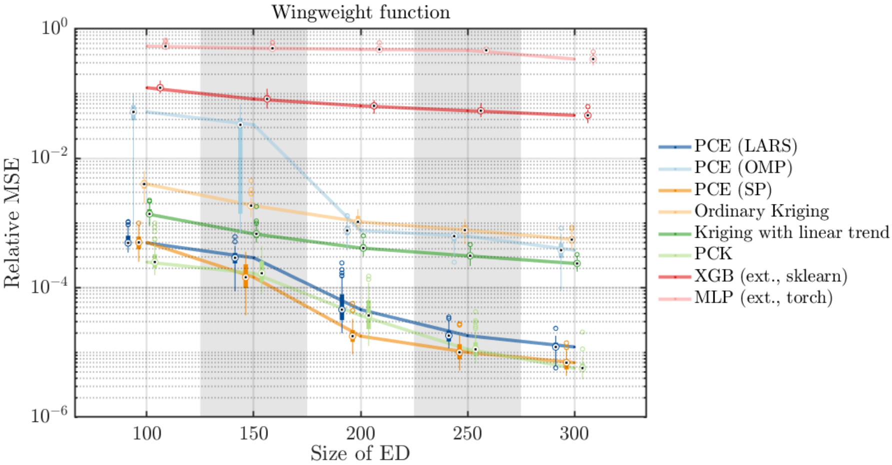
Live demo No. 2: Results

Benchmark analysis post-processing: `uq_display`



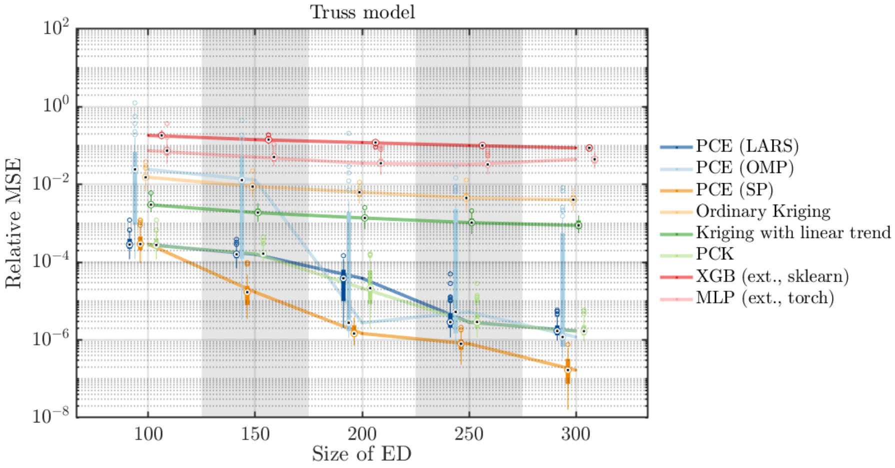
Live demo No. 2: Results

Benchmark analysis post-processing: `uq_display`



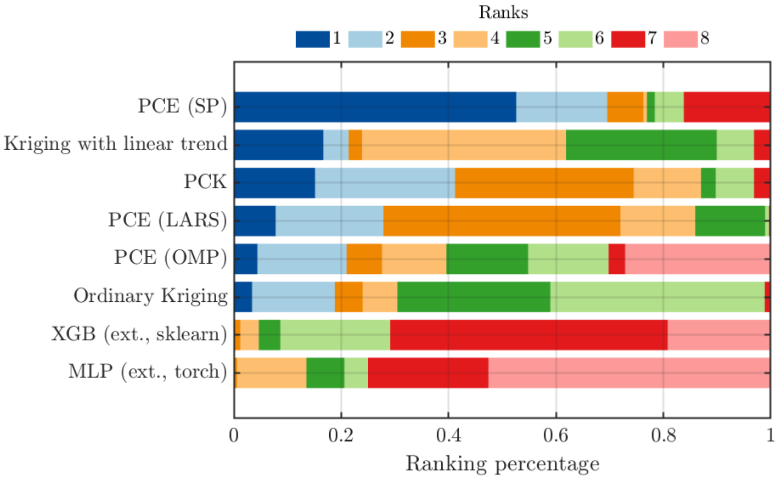
Live demo No. 2: Results

Benchmark analysis post-processing: `uq_display`



Live demo No. 2: Results

Benchmark analysis post-processing: `uq_display`



Live demo No. 2: Results

Benchmark analysis post-processing: `uq_display`