The Journal of Open Source Software

PyUncertainNumber for uncertainty propagation: beyond probability arithmetic

Yu Chen [1¶], **Scott Ferson** [1], **and Edoardo Patelli** [2]

1 Institute for Risk and Uncertainty, University of Liverpool, UK **2** Centre for Intelligent Infrastructure, University of Strathclyde ¶ Corresponding author

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Open Journals

Reviewers:

- @openjournals

Submitted: 01 January 1970
Published: unpublished

Summary

Scientific computations or simulations play a central role in quantifying the performance, reliability, and safety of complex engineered systems. However, these analyses are complicated by the various sources of uncertainties inherent in the computational pipeline. Underestimation may lead to suboptimal performance outside the most common scenarios while overestimation, on the other hand, may lead to over-engineered systems and significant waste of resources. To ensure that complex engineered systems can be operated reliably and robustly, even during rare and extreme environment conditions, a comprehensive analysis is required. The analysis should be comprehensive in two senses: (i) all of the possible sources of uncertainty must be identified and represented using approaite mathematical construct; (ii) that rigorously account for mixed or mixture of various types of uncertainties. One of the biggesdt challenge include xxx, code accessbility, tools to conduct the analysis. By xxx, `pyuncertainnumber` bla bla.. non-intrusively. We interface with many softwares.

Statement of need

A comprehensive uncertainty framework for scientific computation involves a mathematical model, through which various input uncertainties are propagated to estimate the uncertainty of an unknown quantity of interest (QoI). In real-world applications, these input uncertainties are commonly manifested as mixed uncertainties, e.g. probability boxes (p-boxes) which effectively represents a set of distributionbs, combining both the aleatory and epistemic uncertainty in one structure, or a mixture of uncertainties suggesting, for instance, a vector of inputs parameters of aleatory (e.g. probability distributions), epistemic (e.g. intervals), and mixed nature (e.g. probability boxes).

Probability bounds analysis is one of the expressive frameworks proposed to manage uncertainties in an imprecise setting (Beer et al., 2013). Software packages have been developed to facilitate the calculations of uncertain quantities, such as interval arithmetic (Angelis, 2022) and probability arithemetic (A. Gray et al., 2021; N. Gray et al., 2022). Collectively, they can be referred to as *uncertainty arithmetic* (Chen & Ferson, 2025) which straightforwardly computes the response provided the performance function.

While it has the potential to automatically compile non-deterministic subroutines via uncertain primitives, its usages face several challenges, one significant challenge is that code accessibility is often not guaranteed and hence unable to proceed. This would largely restrict the adoption of mixed uncertainty calculations in engineering practice. Such need has been echoed in the engineering applications and also the NASA challenge.

`pyuncertainnumber` addresses that by enabling non-intrusive capability. That is, generic black-box models can be propagated with (that fancy word) various types of uncertainty. This

The Journal of Open Source Software

capability significantly boost its versatility for scientific computations by interfacing with many engineering softwares.

Interval propagation in a non-intrusive manner

Interval analysis has the advantages of providing rigorous enclosures of the solutions to problems, especially for engineering problems subject to epistemic uncertainty, such as modelling system paramters due to lack-of-knowledge or characterising measurement incertitude. Naive interval arithmetic typically faces difficulties such as the infamous interval dependency issue. Though it may be mitigated through mathematical rearrangements in some simple cases, it will be challenging for models of most complex systems. The bigger issue remains the accessibility of code.

Generally, the interval propagation problem can be cast as an optimisation problem where the minimum and maximum are sought via a function mapping. The function, for example f in Equation 1, is not necessarily monotonic or linear and may well be a black-box deterministic model for a generic system.

$$Y = f(I_{x1}, I_{x2}, ..., I_{xn}) \tag{1}$$

where $\mathbf{I} = [\underline{\mathbf{I}}, \overline{\mathbf{I}}] = [I_{x1}, I_{x2}, ..., I_{xn}]^{\mathsf{T}}$ represents the vector of interval-valued inputs. For black box models the optimisation can generally be solved via gradient-free optimisation techniques.

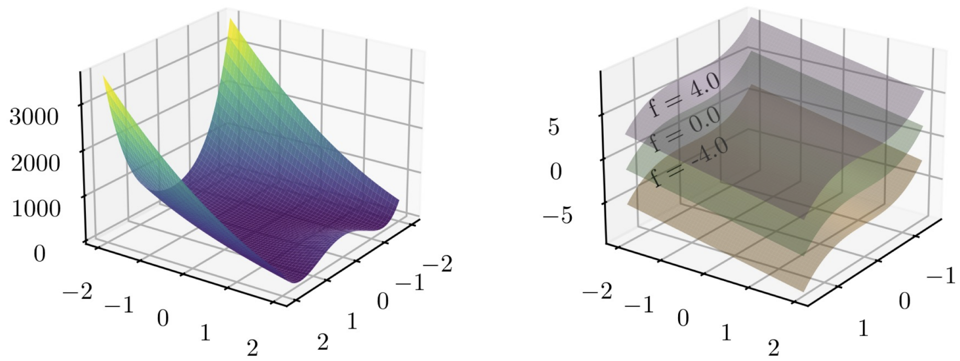
$$\underline{Y} = \min_{\underline{\mathbf{I}} \leq \mathbf{I} \leq \overline{\mathbf{I}}} [f(\mathbf{I})]; \ \overline{Y} = \max_{\underline{\mathbf{I}} \leq \mathbf{I} \leq \overline{\mathbf{I}}} [f(\mathbf{I})]$$

`pyuncertainnumber` provides a series of non-intrusive methodologies of varying applicability. It should be noted that there is generally a trade-off between applicability and computational efficiency. With more knowledge pertaining the characteristics of the underlying function, one can accordinly dispatch an efficient method. For example, when monotonicity is known one can use the vertex method which requres 2^n model evaluations. Furthermore, the accuracy of these methods varies, and a common rule of thumb indicates that increasing the number of model evaluations generally leads to improved accuracy. A summary of applicability is tabulated in **??**, readers can refer to (Chen & Ferson, 2025) for additional details.

Table 1: Several methods for interval propagation

Method	End-points	Subinterval reconstitution	Cauthy-Deviate method	Bayesian optimisation	Genetic algorithm
Assumption	monotonicity	monotonicity in subinterlvas	linearity and gradient required	No	No
Example result	[13.0,148.0]	[13.0,148.0]	[-11.7,100.67]	[13.0,148.0]	[13.0,147.8]

To better demonstrate the non-intrusive capability, two numberical examples, shown below, are provided where they are treated as black-box models. **??** lists the response interval for $f_b([1,5],[7,13],[5,10])$ for associated method.



(a) (b)

Figure 1: Exampler functions as black-box models. (a) $f_a(x, y) = 100(x - y^2)^2 + (1 - x)^2$; (b) $f_b(x, y, z) = x^3 + y + z$

Mixed uncertainty propagation for black-box models

Real complex systems (physical or engineered) of industrial significance typically involves parameters subject to uncertainties of various nature. It requires faithful characterisation of these uncertainties given the empirical information, and the approaach to rigorously progate them. Due to the fact that empirical information is often sparse or scarce or conflicting, even the uncertainty characterisation for one parameter could be of mixed nature, for example one may be confident about the distributional family but uncertain about its shape parameters, or when there exists multiple expert opiontion of different credibility regarding its elicitation. Commonly, real systems expect a high-dimensional input which effectively represents a mixture of aleatory, epistemic, and mixed uncertainties, as symbolfied below:

$$Y = f(\mathbf{u}; C) \tag{2}$$

where $\mathbf{u} \in \mathbb{R}^n$ denotes the collection of n uncertain inputs and C denotes intervariable dependency structure.

When both aleatory and epistemic uncertainties are present in $\mathbf{u}$, a *nested (double) Monte Carlo* approach can be used for determininsitc models without confounding the two distinct types of uncertainty. As illustrated in Figure 2, Latin-hypercube samples are first drawn from the epistemic interval, conditioned on which aleatory samples are drawn from the aleatoric probability distributions. Propagate these samples, which are visually denoted as rug ticks alongside the abscissa, through the computational model results in an ensemble of CDF (cumulative distribution function) of the QoI whereby a final p-box is obtained as the envelope. Each CDF (orange color) correponds to an epistemic sample.

The Journal of Open Source Software

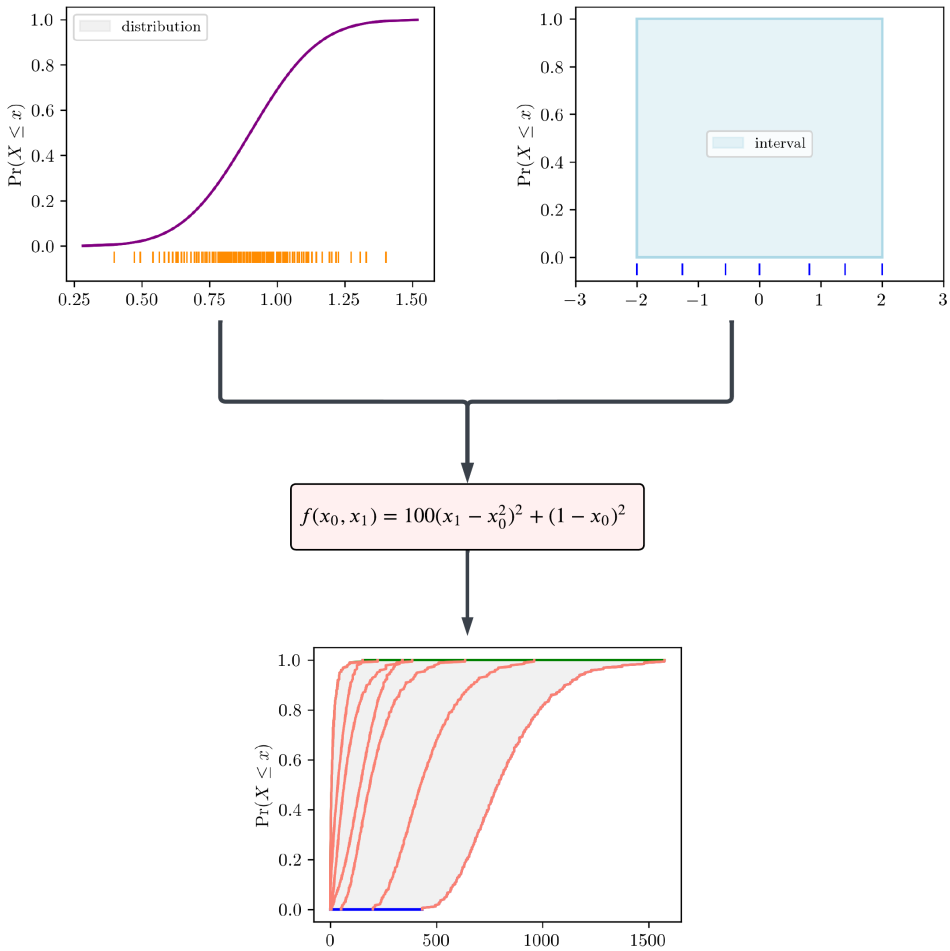


Figure 2: Workflow of the Double Monte Carlo.

To scale to a more realistic setting, Figure 3 illustrates the workflow of *interval Monte Carlo* method where a mixture of aleatory, epistemic, and mixed uncertainty parameters are present, and a certain copula is specified denoting the dependency structure. Correlated samples in the uniform space from the copula, visually denoted as rug ticks alongside the probability axis, are converted to physical space through alpha-cuts. Interval propagation (see the last section) then does the heavy lifting in which scalar values can be considered as degenerate intervals. As a result, the response QoI in Figure 3 is then obtained as a p-box shown in gray. In contrast, a pinched response, obtained from propagating pinched input variables (e.g. a p-box is pinched into a distribution and an interval is pinched into a scalar), is also shown as a comparison. Importantly, rigorous bla bla.

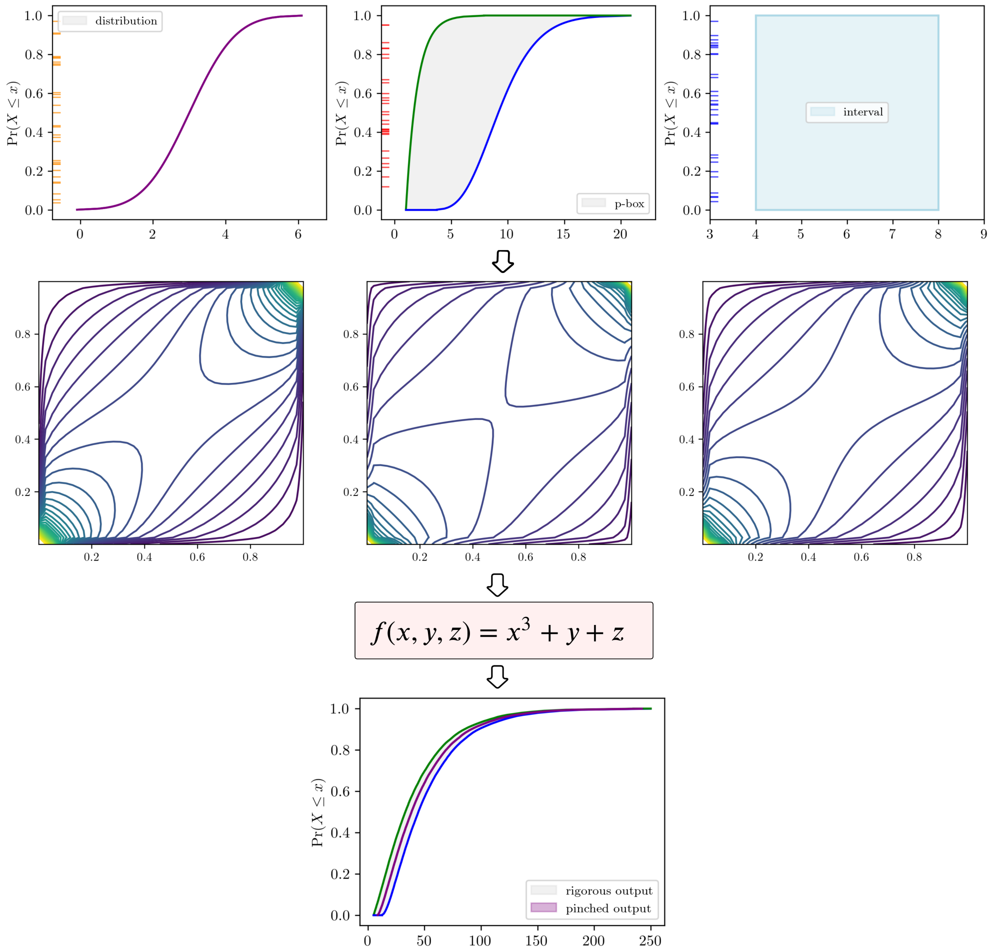


Figure 3: Workflow of the Interval Monte Carlo.

Conclusion

It is evident that computational tasks requiring complex numerical solutions of intervals are non-intrusive (i.e. the source code is not accessiable). `pyuncertainnumber` enables rigorous uncertainty analysis for real-world situations of mixed uncertainties and partial knowledge. Significance: this provides compatability as interfacing with many engineering applications. boost its usage.

Enriched sampling methods bla bla …

Acknowledgements

The work leading to these results received funding through the UK project Development of Advanced Wing Solutions 2 (DAWS2). The DAWS2 project is supported by the Aerospace Technology Institute (ATI) Programme, a joint government and industry investment to maintain and grow the UK's competitive position in civil aerospace design and manufacture. The programme, delivered through a partnership between ATI, Department for Business and Trade (DBT) and Innovate UK, addresses technology, capability and supply chain challenges.

The Journal of Open Source Software

References

Angelis, M. de. (2022). *Intervals* (Version v0.1). Zenodo. https://doi.org/10.5281/zenodo.6205624

Beer, M., Ferson, S., & Kreinovich, V. (2013). Imprecise probabilities in engineering analyses. *Mechanical Systems and Signal Processing*, *37*(1-2), 4–29.

Chen, Y., & Ferson, S. (2025). Imprecise uncertainty management with uncertain numbers to facilitate trustworthy computations. *SciPy Proceedings*.

Gray, A., Ferson, S., & Patelli, E. (2021). ProbabilityBoundsAnalysis. Jl: Arithmetic with sets of distributions. *Proceedings of JuliaCon*, *1*, 1.

Gray, N., Ferson, S., De Angelis, M., Gray, A., & Oliveira, F. B. de. (2022). Probability bounds analysis for python. *Software Impacts*, *12*, 100246. https://doi.org/https://doi.org/10.1016/j.simpa.2022.100246