Open Access | https://doi.org/10.25080/ahrt5264

SciPy 2025
July 7 - July 13, 2025

Proceedings of the 24th
Python in Science Conference
ISSN: 2575-9752

Imprecise uncertainty management with uncertain numbers to facilitate trustworthy computations

Yu Chen[1], and **Scott Ferson**[1]

[1]University of Liverpool

Abstract

Scientific computations of complex systems are surrounded by various forms of uncertainty, requiring appropriate treatment to maximise the credibility of computations. Empirical information for characterisation is often scarce, vague, conflicting and imprecise, requiring expressive uncertainty structures for trustworthy representation, aggregation and propagation. Current practices may present two undesired extremes in terms of uncertainty management, with one endpoint being the total ignorance of uncertainty, whereas the other suggesting overconfidence through the introduction of assumptions unjustified by empirical information. In response to these challenges, this paper demonstrates the framework of `uncertain number`, a unified construct for expressive uncertainty representation at different imprecision. This framework, embedded in the library `pyuncertainnumber`, allows for a closed computation ecosystem whereby trustworthy computations can be conducted intrusively or non-intrusively, hence accomplishing faithful management of uncertainty throughout the computational pipeline. This paper presents an overview of the main capabilities and features of `pyuncertainnumber`.

Keywords uncertain number, imprecise probability, probability bounds analysis, optimisation under uncertainty

1. Introduction

Large-scale scientific computing applications for complex physical and engineered systems are often approximate and uncertain, notably due to limited knowledge of the physical processes, approximate representation of the model, and sparse experimental data for calibration, etc [1], [2], [3]. Besides, engineered systems are often required to operate safely and robustly under varying environments or uncertain operational conditions, requiring uncertainty to be comprehensively considered in order to maximise the credibility of predictions, designs and decisions [4].

It is vital to *know what you do not know* in terms of trustworthy modelling and predictions, suggesting that *all models are wrong but some are useful* on the condition of knowing their assumptions and applicability, hence building the credibility of the computational results. It is often a challenge, in modelling complex physical phenomena, to construct mathematical models in a quantitative manner, on one hand, without ignoring significant information and, on the other hand, without introducing unwarranted assumptions [5], [6]. The bottleneck is usually the limited information in terms of both knowledge and experimental data.

The increasing awareness of the differentiation of aleatory and epistemic uncertainty arises the need for more expressive mathematical frameworks to reason with various forms of uncertainty [7], [8]. Imprecise uncertainty frameworks [9], such as evidence theory, random set, possibility distributions, credal set, capacities are proposed to reflect the situations

Published Jul 10, 2025

Correspondence to
Yu Chen
lesliechen666@outlook.com

when information is scarce, vague, conflicting or imprecise whereby precise distribution are hard to be defined. Given the available information, there exist two untenable extreme (exclusive) endpoints that prescribe an interval of trustworthiness. The lack of uncertainty quantification presented in many deterministic numerical simulations constitutes the lower bound, whereas the overconfidence of the computation, through introduction of unwarranted assumptions about uncertainties not faithful to the state of knowledge, as the upper bound.

We aim at a faithful management of uncertainty throughout the computational pipeline using `uncertain number`, a unified construct for uncertainty characterisation at different imprecision. This paper demonstrates the framework of `uncertain number` which allows for a closed computation ecosystem whereby trustworthy computations can be conducted in a rigorous manner. This paper presents an overview of the main capabilities of the library `pyuncertainnumber`[1]. To facilitate clearer illustration and reproducibility of the workflow, essential code snippets are included in the main text, while a comprehensive tutorial is available in the Supporting Documents section.

2. Expressive power of uncertain number

When characterising input parameters to a simulation model, while probability distribution has long been used to reflect uncertainty (*variability*[2]), there is still uncertainty (*incertitude*[3]) about the distribution shape and interdependencies. Empirical information varies in quality and nature, which may be quantitative, qualitative and even linguistically vague. Multiple sources or elicitation could vary in credibility and even be conflicting. Empirical data, if any, may be scarce or imprecise due to the inaccuracy of experimental measurements, or prohibitive cost of collecting data. It is therefore challenging to formulate suitable uncertainty models given partial information without introducing unwarranted assumptions. To this end, more expressive mathematical constructs are needed.

`Uncertain number` stands for a generalised representation that unifies several uncertainty constructs including intervals, probability distributions, probability boxes (p-boxes) and Dempster-Shafer structures (DSS), plus real numbers. These constructs are closely related to each other: an interval indicates a value imprecisely known within a range where no assumptions of likelihood about the enclosed values are made; P-boxes can be considered as interval bounds on cumulative distributions and a DSS can be deemed as a discrete distribution with interval quantiles. Parametric p-boxes $F_X(x \mid \theta^I)$ are probability distributions whose parameters θ^I and samples are intervals, and an interval ($I = [a, b]$) can be identified as a p-box $[H_a(x), H_b(x)]$ whose bounds are unit step functions denoted by $H(x)$; a p-box can be discretised into a DSS with pairs of intervals (focal elements) and probability masses $\left\{([a_i, b_i], p_i)_1^N\right\}$, and conversely a DSS can be stacked with a list of intervals. Importantly, all of these constructs are special cases of free p-boxes which effectively represents a set of distributions.

Figure 1 visually illustrates the notion of an `uncertain number`, which is underpinned by a probability bounding approach [10], [11], [12] that allows for a faithful representation of the state of empirical information. For example, it can be characterised as a real number (a *degenerate* of an interval) when there is no uncertainty, a precise distribution (a *degenerate*

[1] `pyuncertainnumber` is a research-grade open-source library, with its source code available on GitHub and comprehesive documentation available online.

[2] also called randomness, aleatory uncertainty, objective uncertainty, dissonance, or irreducible uncertainty arised from natural stochasticity, environmental or structural variation across space or through time.

[3] also called ignorance, epistemic uncertainty, non-specificity, or reducible uncertainty arised from incompleteness of knowledge.

of a p-box) when there is abundant data, and a set of distributions (e.g. a p-box) when there is partial information.

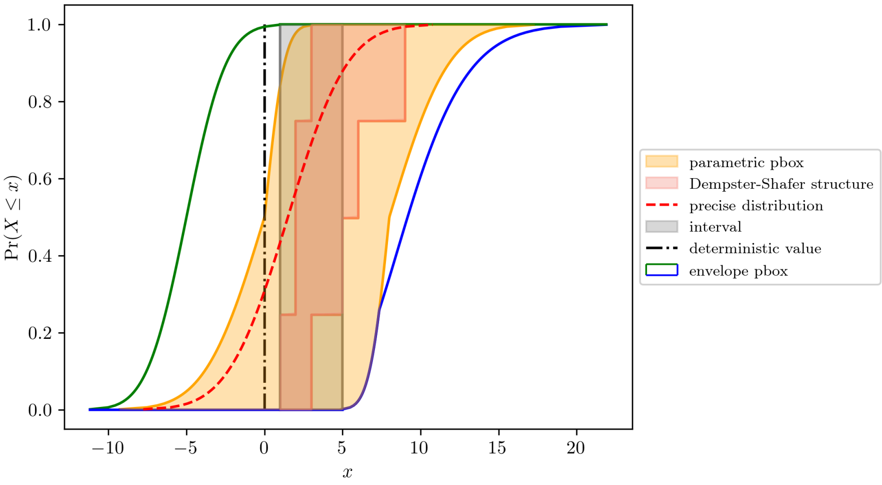

Figure 1. *Uncertain number refers to a generalised representation that unifies several constructs including intervals, probability distributions, probability boxes (p-boxes) and Dempster-Shafer structures, plus real numbers.*

2.1. *A probability bounding approach on information constraints*

Often there is little empirical information pertaining some parameters of a mathematical model. A faithful characterisation entails that all of the available statistical information should be utilised but without introducing any extra assumptions beyond what are empirically justified.

The bounding approach presents as a natural reflection of the state of epistemic uncertainty, which tightens the bounds with extra empirical information. As shown in Figure 2, the level of information specifies constraints to accordingly construct the uncertain number, which starts as little as a coarse estimated range, to moment information, further to the knowledge of shape such that uncertainties are pinched to a precise distribution. These bounds are best-possible[4] in the sense that they could not be any tighter without excluding CDF satisfying the specified constraints, or without additional information [7].

Conveniently, `pyuncertainnumber` provides a bespoke constructor to facilitate the faithful characterisation of uncertain quantities based on known information, which includes limits on quantiles, information about summary statistics such as mean, mode or variance, and qualitative information about distribution shape, such as whether it is symmetric or unimodal.

Figure 2. *Illustration of the idea of level of information specified as constraints*

[4] the uncertain number could not be any tighter without more information.

```
import pyuncertainnumber as pun

# specify available empirical information as constraints
pun.known_constraints(minimum=0, maximum=2., mean=1, var=0.25)
```

2.2. *Aggregation of uncertainty*

One of the controversial subject in uncertainty analysis is the aggregation of multiple (imperfect) sources of information, evidence, or expert elicitations. Information varies in quality and could be conflicting. Experts may have different degrees of subjectivity and different representations for the uncertain quantity, leading to epistemic uncertainty in the various educated estimates.

`Uncertain number` presents one advantage that it provides a unified structure to enclose the aggregation operation on a set of expert elicitations, whatever the forms they may be, whether a distribution or an interval. Figure 3 shows an example of the elicitation from a couple of experts with their credibility shown as probability masses. This explains further the provenance of the constructs in Figure 1. `pyuncertainnumber` supports several aggregation rules (e.g. envelope, mixture, and intersection, etc) enabling diverse level of information to be pooled stochastically or conservatively.

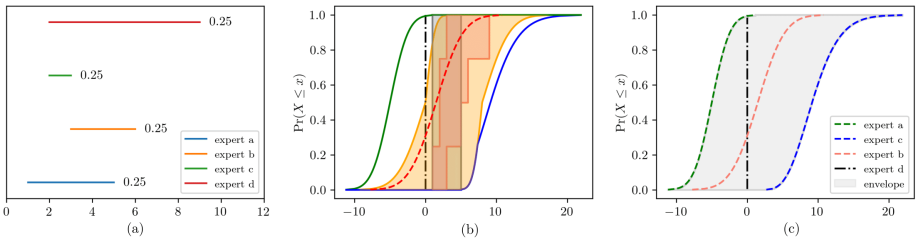

Figure 3. *Uncertainty aggregation: **(a)** expert knowledge expressed as intervals coupled with their credibility; **(b)** the same uncertain number in Figure 1 of which the DSS is composed by the aggregation in (a), and the circumscribed p-box is composed by the envelope in (c); **(c)** expert elicitation as precise distributions whereby an envelope operation is taken*

```
from pyuncertainnumber import stochastic_mixture, envelope

""" note aggregation operations apply to all constructs of uncertain numbers """

# mixture aggregation of interval eliciion into a Dempster-Shafer structure
a = pun.I(1, 5)  # b, c, d ...
mix = stochastic_mixture(a, b, c, d, masses=[0.25, 0.25, 0.25, 0.25])

# envelope aggregation of distributional eclicitation into a p-box
a = pun.D('gaussian', (-5, 2))  # b, c, d ...
env = envelope(a, b, c, d)
```

2.3. *Measurement imprecision*

Empirical data rarely come in perfect forms, especially for *in situ* measurements. Practical computations frequently deal with poor measurements with different imprecision, possibly arising in recording, transmission, communication or manipulation, etc [13]. Intervals turn out to be natural constructs for representing *incertitude* in imprecise measurements, manifested either in a direct interval or a plus-or-minus form [14]. When interval-valued measurements are present in a data set, a single probability distribution is inadequate to characterise the epistemic uncertainty. Rather, the bounding strategy applies whereby classical inference methods are extended to both characterise the sampling uncertainty and also data imprecision. For example, as shown in Figure 4, a set of maximum likelihood estimates $\lambda^I = [\underline{\lambda}, \overline{\lambda}]$ are yielded for a single datum where λ indicates the parameter of exponential distribution. Collectively, the dataset leads to an interval bound of the fitted exponential distribution shown in the shaded area in Figure 4, along with the true data generating mechanism (DGM) in red dotted curve and the empirical CDF in gray dotted curve. Further, as a nonparametric comparison, the Kolmogorov Smirnov confidence bands is also extended [15], as shown in blue dashes below. These are both rigorous uncertain numbers that enclose the true yet unknown data generating distribution.

$$[\underline{F}(x), \overline{F}(x)] = \left[\min\left(1, \hat{F}_L(x) + D_N^\alpha\right), \max\left(0, \hat{F}_R(x) - D_N^\alpha\right)\right] \tag{1}$$

where $\hat{F}_L$ and $\hat{F}_R$ represent the empirical CDF on endpoints of the imprecise dataset, and D_N^α denotes the Smirnov critical value at significance level α.

Figure 4. *Characterisation of imprecise measurements. **(a)** fitting an exponential distribution to an interval datum by maximum likelihood estimation; **(b)** Parametric and nonparamettric characterisation of the whole imprecise data set $\{x_i\}$ which includes 15 data points i.i.d (independent and identically distributed) sampled from an exponential distribution (i.e.* $\mathrm{Exp}(0.4)$*) contaminated by a margin of error* $\Delta = 1.4$*. That is,* $x_i = [\underline{x}_i, \overline{x}_i] = [m_i - \Delta, m_i + \Delta]$ *where* $m_i \sim \mathrm{Exp}(0.4)$.

```python
import scipy.stats as sps
from pyuncertainnumber import pba

# synthetic generation of the imprecise data
precise_sample = sps.expon(scale=1/0.4).rvs(15)
imprecise_data = pba.I(lo = precise_sample - 1.4, hi=precise_sample + 1.4)

# parametric distributional estimator using method of matching moments
pun.fit('mom', family='exponential', data=imprecise_data)

# nonparametric estimator using Kolmogorov Smirnov confidence bands at 95% confidence level
pun.KS_bounds(imprecise_data, alpha=0.025, display=True)
```

2.4. *Linguistic numerical hedges for uncertainty interpretation*

Minimally, qualitative linguistic description may be used to express the estimates over numerical input values. Those are called numerical hedges, which may include colloquial words such as “about”, “around”, “almost” etc. [16]. With the focus on NLP in many deep learning applications in recent years, the interpretation of hedged words and their quantitative implication of uncertainty is vital for downstream safety-related applications. `pyuncertainnumber` provides support to interpret these hedges in an effort to build a consistent basis for consistent uncertainty elicitation and communication in situations where empirical information is almost minimal. Importantly, real numbers such as “7” or “7.0” can also be interpreted as an interval based on significant digits [16], potentially leading to a complete system of rigorous numerical numbers.

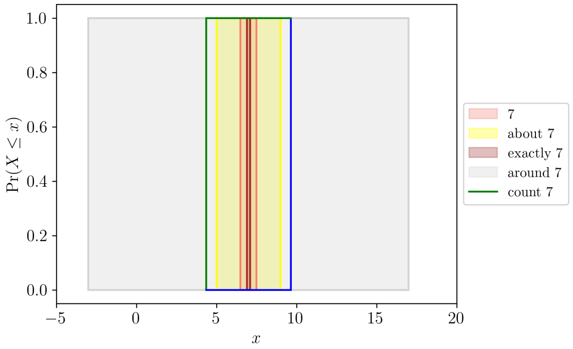

Figure 5. *Illustration of numerical hedges.*

```
pun.hedge_interpret('about 7')  # similar usage for other hedges, such as '7.0'
```

2.5. Dependency structure: fully specified, partially known or unknown

Neglecting the incertitude about the dependency structure and assuming independence anyway constitutes a methodological bad practice. Maybe the most notable example of misuse of dependency structure is the 2008 financial crisis [17].

Similar as the bounding approach applies onto marginal distributions in the face of various levels of information, the inter-variable dependency information can also be known, partially known or unknown within the probability bounding framework. Collectively, it manifests the imprecise specification of a joint distribution. As an example, Figure 6 illustrates the interval bounds of a joint CDF of a bivariate p-box. In this case, sampling operations will produce bivariate interval constructs, shown as the rectangles at the bottom of Figure 6. These constructs can all be consistently represented as uncertain numbers. Additional details regarding the imprecise dependency specification will be discussed in Section 6.

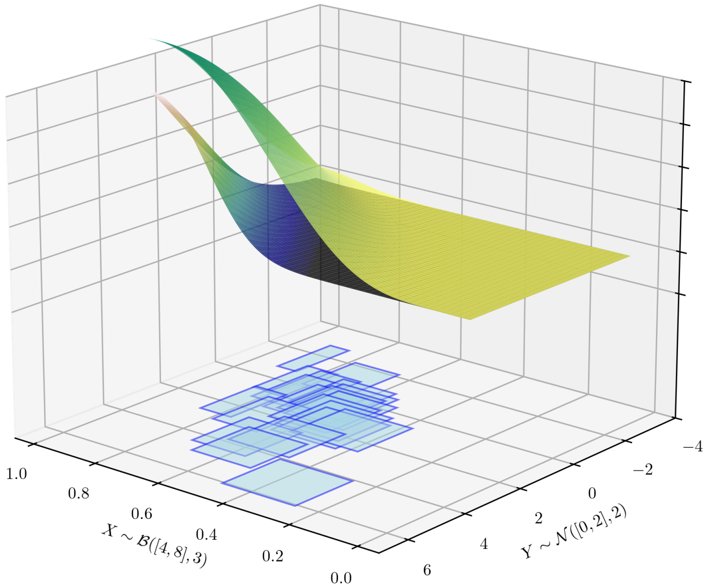

Figure 6. *A bivariate p-box with marginals $X \sim \mathcal{B}([4,8],3)$ and $Y \sim \mathcal{N}([0,2],2)$, and Gaussian copula parameterised by $\rho_{XY} = -0.8$. The upper of the plot displays the joint CDF with upper and lower surface and the bottom shows the sampled deviates which are bivariate intervals*

3. Non-deterministic computations and uncertainty propagation

Scientific computations are desired to be *rigorous* and *best-possible* when subject to uncertainties.

3.1. *Probability bounds analysis*

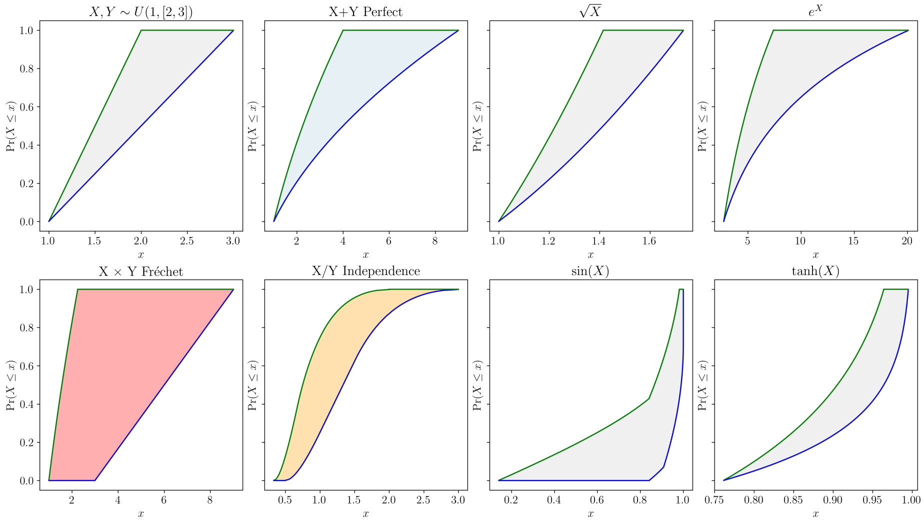

Figure 7. *Illustration of p-box arithmetic under various dependency structure. The left four show different dependency structures in colors with different binary operations, the right four show differnt unary operations.*

Probability bounds anlaysis (PBA) combines both interval analysis and probability theory, allowing rigorous bounds of (arithmetic) functions of random variables to be computed even with partial information [7], [11], [12]. Intuitively, as interval arithmetic enables rigorous calculation for sets of real numbers, PBA accomplish the same for sets of distributions. P-box arithmetic is built upon generalised probability convolutions, which through further extensions cover a wide spectrum of arithmetic operations: unary transformations, binary operations between p-boxes or Dempster-Shafer structures, and general functions composed of a series of base operations. It also covers a wide spectrum of dependency structures which could be fully known (specified copula C), partially known (lower bound copula $\underline{C}$), or even unknown (Fr ' e chet bounds [18]).

Importantly, we provide a unified interface and implementation through `pyuncertainnumber`, enabling rigorous calculations regardless of the representation of the uncertain variables or their dependency structure. This interface has abstracted out much of the arcane details in terms of conversion, discretisation, condensation, and stacking such that analysts can benefit the consistency and focus more on the effects the calculations.

Notably, the Fr ' e chet bounds provides a trustworthy default when no dependency information is known, in that the resulting bounds rigorously enclose the result no matter what correlation or nonlinear dependency may exist between these variables. This faithfully translates the situation of unknown into making no assumption at all, as opposed of an unjustified independence assumption. Equation (2) gives the Fr ' e chet expression [18] for the addition of uncertain numbers and multiplication and division have similar forms, where $[\overline{F}_X(x), \underline{F}_X(x)]$ denotes the interval bounds of CDF of the composing uncertain number.

$$\left[\, \overline{F}_{X+Y}(z), \underline{F}_{X+Y}(z) \,\right] = \left[\inf_{z=x+y} \min\left(\overline{F}_X(x) + \overline{F}_Y(y), 1 \right), \sup_{z=x+y} \max(\underline{F}_X(x) + \underline{F}_Y(y) - 1, 0) \right] \quad (2)$$

Figure 7 displays several common arithmetic examples. These arithmetic operations can be performed in the same signature in `pyuncertainnumber` as real number arithmetics in Python via operator overloading. Fr ' e chet are employed as default except the dependency can be otherwise justified. Importantly, now these calculations yield regirous results that are guaranteed to enclose all possible distributions of the output variable so long as the input uncertain number were all sure to enclose their respective distributions.

3.2. *Non-deterministic propagation*

Scientific computing typically involves a mathematical model, for example a coupled system of nonlinear partial differential equations, to simulate the behaviour of natural or engineered systems. Many such applications involve high-fidelity numerical solutions as a complicated black-box model (e.g. CFD) in a non-intrusive setting.

For a comprehensive uncertainty analysis, all uncertainty sources (e.g. model inputs, initial or boundary conditions, model form assumptions, numerical approximation, and model extrapolation) should be appropriately characterised and and have their contributions to the total uncertainty of the system response quantity of interest (QoI) elucidated, such that efficient uncertainty reduction or management can be conducted by decision makers. A notable example that embodies the idea of such uncertainty framework is the NASA UQ challenge [19].

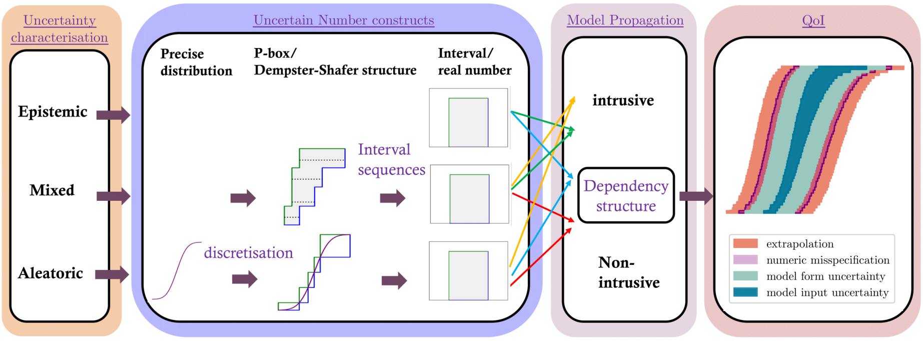

Figure 8. *Flowchart of a comprehensive uncertainty analysis pipeline. Various dependency structures are denoted in colored arrows.*

Figure 8 outlines a general computational pipeline typical for scientific numerical computations with a black-box model, which features the probability bounding approach during the stages of characterisation, propagation and finally the representation of QoI, and echos the idea that aleatory and epistemic uncertainty should be treated differently. Pairs of bounds constitute layered colour-shaded regions reflecting the sources of uncertainties involved. This provides a straightforward and intuitive way to account for the contributions of uncertainties in a comprehensive framework.

It should be noted that *interval analysis* plays a central role in the propagation of epistemic uncertainty and further mixed uncertainty or mixture of uncertainty. The *slicing* algorithm discretises a p-box into intervals and effectively propagate intervals. The same rigorous strategy also applies to precise distributions through outward-rounding.

Regarding the propagation methods for different types of uncertainty, various methods have been proposed, intrusive or nonintrusive, depending on the characteristic of the performance function such as linearity, monotonicity, or the accessibility to gradients. Table 1 lists the characteristics of many methods. Similar to the cases of information constraints for uncertainty characterisation, the more knowledge about the mathematical model, the higher chances of finding an efficient method. Notably, enriched sampling methods such as nested Monte Carlo or interval Monte Carlo allows the outward (rigorous) implementation for a sampling signature consistent with aleatory uncertainty. This demonstrates the universal applicability and compatibility of sampling-based approaches for generic deterministic black-box models in scientific computing. `pyuncertainnumber` provides high-level API to these methods and will automatically inspect the appropriateness of the propagating method given the characteristics of the function and uncertainty.

```python
""" a typical workflow of uncertainty characterisation and propagation """
from pyuncertainnumber import Propagation

# constructions of uncertain number
a = pun.I(2, 3)
b = pun.normal(4, 1)
c = pun.uniform([4,5], [9,10])

# specify a response function
def foo(x): return x[0] ** 3 + 5 * x[1] + x[2]

# intrusive signature where uncertain numbers are drop-in replacements
response = foo([a, b, c])

# alternatively, to use a generic high-level propagation API
p = Propagation(vars=[a,b,c],
      func=foo,
      method='slicing',
      interval_strategy='direct'
)

# heavy-lifting of running propagation
response = p.run(n_slices=50)
```

Table 1. *Characterisitics of uncertainty propagation methods*

uncertainty	Methods	cost	characteristics	Non-intrusive applicability
Aleatory	Monte Carlo methods	***	Brute-force sampling	Yes
	Taylor expansion	*	linearity assumption; gradient required	No
Epistemic	interval arithmetic	*	vulnerable to dependency problem	No
	Endpoints	**	monotonicity over intervals; exponential growth 2^d	Yes
	Subinterval reconstitution	***	computationally intensive with exponential growth $(m+1)^d$	Yes
	Cauthy-Deviate method	***	linearity assumption	Yes
	gradient-based optimisation	*	gradients required	No
	Gradient-free optimisation	***	general applicability	Yes
Mixed	Slicing	***	transformed into interval propagation; cartesian product required	Yes
	interval Monte Carlo	***	general applicability with heavy cost; cartesian product required	Yes
	nested Monte Carlo	***	double-loop sampling conditioned on epistemic variables; mixture of uncertainty	Yes

3.3. *Uncertainty-aware surrogate models*

Modern advanced numerical simulations are often computationally expensive, making abundant model evaluations required by uncertainty analysis impractical or even intractable. Surrogate models are therefore utilised to learn and generalise from observed data (a limited subset of DOE). Epistemic uncertainties exist not only in the model form (i.e. parameter and structure) but also in the extrapolation of predictions.

To account for such uncertainty, a probabilistic take of machine learning brings models that represent parameters in probability distributions (Bayesian neural networks), and models that represent a distribution of function structures (Gaussian Process). On the other hand, an intervalised or distribution-free view employs models that consider interval-valued parameters (Interval Predictor Model). These models have the prospects of efficiently propagating the input uncertainties to fully characterise the tail probability of the QoI for example in the imprecise reliability analysis (will be discussed in Section 9).

With many machine learning frameworks available in the Python ecosystem (e.g. Tensorflow, GPflow, PyIPM, etc), we provide the algorithms and the interface in `pyuncertainnumber` to extend those models to be combined within our framework to further propagate uncertain numbers for a efficient and comprehensive uncertainty management.

4. Risk, reliability, and design optimisation under uncertainty

Uncertainty is often a means to an end for scientific computations. For example, risk assessment of environmental hazards (e.g. earthquakes), evaluation of performance and reliability of engineered system under uncertain environmental conditions, and supports of decision-makings for optimal engineering designs. An increasing acknowledgement of the mixture of uncertainty encountered in real world problems has led to the transition from the deterministic safety factor approach to probabilistic risk assessment, further to performance-based or reliability-based design protocols in several safety-critical industries [5]. A comprehensive and expressive uncertainty framework that reflects the contributions of various uncertainties is essential, and an accessible toolkit with universal uncertainty representations is beneficial in facilitating the adoption of comprehensive uncertainty management.

4.1. *Imprecise reliability analysis*

Conventionally, in the analysis of a probabilistic safety framework, the probability of failure is given as $p_f = \boldsymbol{P}[g(\boldsymbol{X}) \leq 0]$ which involves an integration of the joint probability distribution over the failure domain. Given the presence of epistemic uncertainty, the failure probability presents as an interval bound as opposed of precise probability measure. Consider a generic simulation in the face of mixed uncertainties such that the system variables $\boldsymbol{X} = (X_1, ..., X_n)^T$ cannot be precisely characterised but to be more appropriately modelled as an `uncertain number`, as suggested in Expressive power of uncertain number. The interval of failure probability can be given as [20]:

$$\left[\underline{p}_f, \overline{p}_f\right] = \Big[\sum_{\eta_i : \geq \inf(g(\eta_i))} \alpha_i, \sum_{\eta_i : \geq \sup(g(\eta_i))} \alpha_i \Big] \tag{3}$$

in which $\inf$ and $\sup$ denote the infimum and supremum function; Each uncertain number X_i is discretised as pairs of focal elements and probability masses $\left\{(\eta_i, \alpha_i)_1^N\right\}$. The computation of $g(\boldsymbol{X})$ can be straightforwardly done by the methods discussed in Section 7 and `pyuncertainnumber` provides a simple function to easily evaluate the probability interval.

Figure 9 illustrates the conceptual comparison of expressing the probability of failure for both probabilistic and imprecise frameworks. Conventionally, the p_f can be estimated using a Monte Carlo estimator that reads: $p_f = \frac{1}{N}\sum_i^N \boldsymbol{I}_{g(\boldsymbol{X})\leq 0}(\boldsymbol{x}_i)$, where $\boldsymbol{I}$ denotes an indicator function, as shown in different colours in the histogram. This summation also depicts the blue cross in the empirical CDF in the top figure, where an interval of $p_f \in \left[\underline{p}_f, \overline{p}_f\right]$ is shown in red.

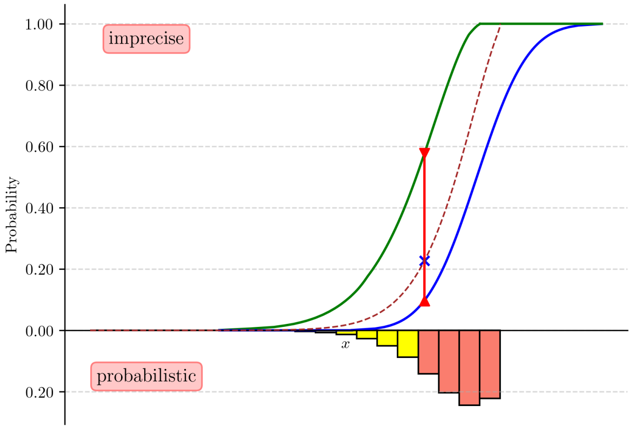

Figure 9. *Illustration of failure probability $\mathbb{P}[g(\boldsymbol{X}) \leq 0]$ for both probabilistic and imprecise frameworks where the probabilistic input is enclosed in the imprecise input*

4.2. Robust designs under epistemic and mixture of uncertainty

Considerable progress has been done towards design optimisation under uncertainty with a probabilistic framework. Typical formulations include robust optimisation aiming for designs less sensitive to variation and reliability-based optimisation whose constraints are less prone to failure under variability. However, real world scenarios commonly involve lack-of-knowledge uncertainties and, more often, mixture of uncertainties, which brings additional subtleties in the formulation. In recognition of the epistemic uncertainty, optimisation is directed with respect to an interval bound. As an example, Figure 10 shows a simple exemplar objective function assumed to represent the cost of manufacture, see (4), in the face of pure epistemic uncertainty.

$$C(x1, x2) = x_1 x_2 - \sin(x_1) x_2^2 + x_1^2 + 300 \tag{4}$$

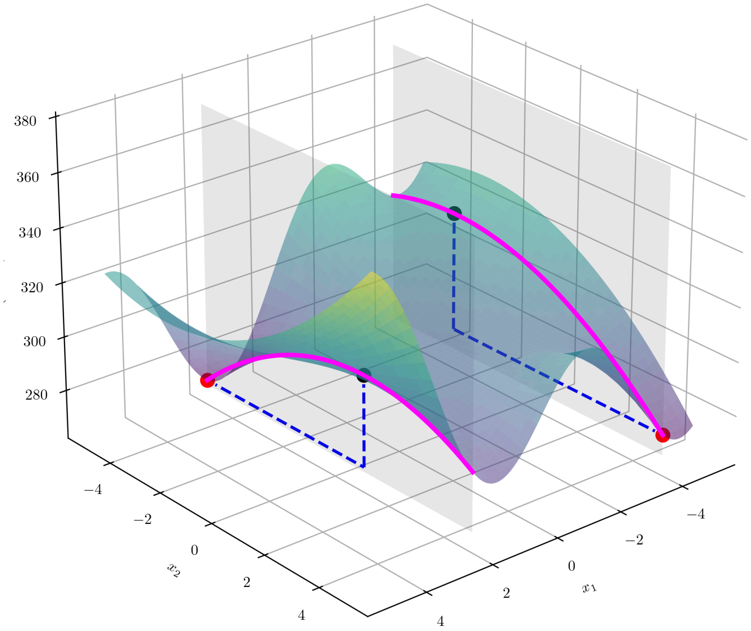

Figure 10. *An examplar objective function. Two vertical planes corresponding to design variables $x_1 = -4$ and $x_1 = 2$ are shown, in which the intersecting curves as well as the maximum and minimum are specifically marked.*

where x_1 denotes design variables of the range of $[-5, 5]$, while x_2 denotes uncertain parameters. Given the pure epistemic uncertainty, a performance-oriented objective can be intuitively formulated as minimising the upper bound: $\min \overline{C}$, or a robustness-oriented objective which minimises the width the cost interval: $\min w\left(\left[\underline{C}, \overline{C}\right]\right)$.

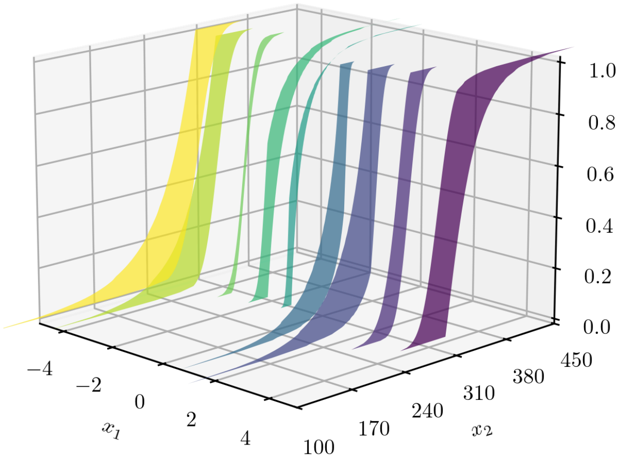

Figure 11. *Illustration of the cost objective represented in uncertain numbers with respect to designs (x_1) in the face of mixed-type uncertain parameter (x_2).*

In mixed uncertainty cases, the cost w.r.t a design is effectively presented by p-boxes , as seen in Figure 11. The existence of epistemic domain results in a bounded objective such as an expected performance indicator or a probability measure, for which both operations will average out the aleatory space. Intuitively, in this example, one would desire to find the design accomplishing the minimum of the expected cost $\left[\underline{\mathbb{E}}(C), \overline{\mathbb{E}}(C)\right]$. Alternatively, it is also possible to prescribe the performance requirement in a probability statement as below, which optimises with the probability of having a cost higher than a prescribed budget c_j, where $p_f(C) = \mathbb{P}\left[C(x_1, x_2) > c_j\right]$.

$$x_1^* = \arg\min \overline{\mathbb{E}}(C) \text{ or } x_1^* = \arg\min \overline{p}_f(C) \tag{5}$$

This effectively introduces a constraint as to the budget. Note that upper bounds are used in the current formulation to take into effect of the epistemic uncertainty and to reflect a sense of performance guarantee. Additionally, to be more pragmatic, the objective can be compounded with another constraint which could for example reflects a cost-production tradeoff that a lower cost is aimed on the condition that a certain production requirement is probabilistically satisfied. An examplar formation can be given as follows where g_i stands for extra constraint functions:

$$\arg\min\left\{\overline{\mathbb{E}}(C) : \mathbb{P}[g_i(x1, x2) \leq 0] \geq 1 - \varepsilon_i\right\} \tag{6}$$

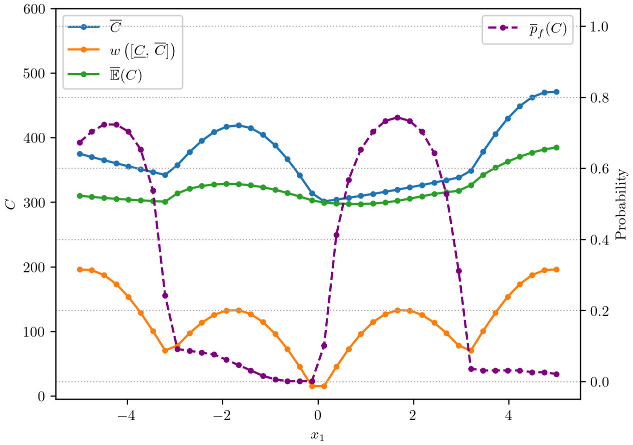

Figure 12. *Optimisation tasks of different formulations in the face of mixed uncertainty*

5. Conclusion and Outlook

To know what you do not know suggests the importance to realise the assumptions and applicability of scientific computations to maximise the credibility for the resulting predictions, designs and decisions. Given the various sources of uncertainties and the often limited empirical information, how to appropriately represent, aggregate, propagate uncertainties is a critical challenge for trustworthy reliability and risk assessments, especially for safety critical applications. Current practices tend to go overboard with unjustified assumptions of Gaussianality and independence, mostly due to the computational simplicity and the lack of tools for a comprehensive uncertainty analysis. This leads to a need of a computational framework that effectively balances representational expressiveness and computational feasibility.

This paper presents the framework of `uncertain number` which fills this gap and presents several advantages: (i) it is highly expressive enabling faithful characterisation of an uncertain quantity given various scenarios of partial knowledge where conventional probability theory struggles to cope; (ii) it provides a closed environment where various characterisation, aggregation and propagation operations can be consistently conducted based on the unified structure, `uncertain number`; (iii) it is underpinned by a probability bounding mechanism which intuitively showcases the notion of epistemic uncertainty — wherein increased knowledge leads to progressively tighter bounds; (iv) the bounding mechanism enables an explicit differentiation of aleatory and epistemic uncertainties, allowing their respective contributions to be accounted for during both characterisation and propagation.

The developed Python library, `pyuncertainnumber`, facilitates trustworthy management of uncertainty through faithful representation and rigorous propagation. Unlike other established Python-based general purpose uncertainty quantification (UQ) tools that primarily focus on probabilistic modelling —– often requiring precise specification of distribution shape and dependency structures, which is challenging in practice in the face of partial information, `pyuncertainnumber`, however, relaxes such specification and is capable of rigorous treatment even with unknown knowledge states. It explicitly addresses both aleatory and epistemic uncertainty using appropriate methods without making unjustified assumptions, advancing uncertainty management to handle both variability and incertitude. Consequently, it enables a more robust framework for addressing real-world UQ challenges such as mixed uncertainty propagation and characterisation under partial knowledge.

As probabilistic programming provides support for automatic inference, we aim for an imprecise uncertainty analysis framework, where variables are consistently represented by uncertain numbers in the face of both variability and incertitude, allowing for extensions of deterministic functions to be computed in an automatic, comprehensive, and rigorous manner. Our next focus will be extending the framework of probabilistic programming, which has a heavy focus on Bayesian inference, into an imprecise realm where more comprehensive uncertainty structures can be integrated into the learning and inference of machine learning models.

Acknowledgement

This work has been supported by the DAWS2 (Development of Advanced Wing Solution 2) project funded by Innovate UK.

References

[1] C. J. Roy and W. L. Oberkampf, “A comprehensive framework for verification, validation, and uncertainty quantification in scientific computing,” *Computer methods in applied mechanics and engineering*, vol. 200, no. 25–28, pp. 2131–2144, 2011, doi: https://doi.org/10.1016/j.cma.2011.03.016.

[2] W. L. Oberkampf, J. C. Helton, C. A. Joslyn, S. F. Wojtkiewicz, and S. Ferson, “Challenge problems: uncertainty in system response given uncertain parameters,” *Reliability Engineering & System Safety*, vol. 85, no. 1–3, pp. 11–19, 2004, doi: 10.1016/j.ress.2004.03.002.

[3] R. C. Smith, *Uncertainty quantification: theory, implementation, and applications*. SIAM, 2024. doi: 10.1137/1.9781611973228.

[4] A. Gray *et al.*, “From inference to design: A comprehensive framework for uncertainty quantification in engineering with limited information,” *Mechanical Systems and Signal Processing*, vol. 165, p. 108210, 2022, doi: 10.1016/j.ymssp.2021.108210.

[5] M. Beer, S. Ferson, and V. Kreinovich, “Imprecise probabilities in engineering analyses,” *Mechanical systems and signal processing*, vol. 37, no. 1–2, pp. 4–29, 2013, doi: 10.1016/j.ymssp.2013.01.024.

[6] E. Patelli, M. Broggi, S. Tolo, and J. Sadeghi, “COSSAN SOFTWARE: A MULTIDISCIPLINARY AND COLLABORATIVE SOFTWARE FOR UNCERTAINTY QUANTIFICATION,” in *Proceedings of the 2nd International Conference on Uncertainty Quantification in Computational Sciences and Engineering (UNCECOMP 2017)*, in UNCECOMP 2017. 2017, pp. 212–224. doi: 10.7712/120217.5364.16982.

[7] S. Ferson and J. G. Hajagos, “Arithmetic with uncertain numbers: rigorous and (often) best possible answers,” *Reliability Engineering & System Safety*, vol. 85, no. 1–3, pp. 135–152, 2004, doi: 10.1016/j.ress.2004.03.008.

[8] S. Ferson and L. R. Ginzburg, “Different methods are needed to propagate ignorance and variability,” *Reliability Engineering & System Safety*, vol. 54, no. 2–3, pp. 133–144, 1996, doi: 10.1016/s0951-8320(96)00071-3.

[9] S. Destercke, D. Dubois, and E. Chojnacki, “Unifying practical uncertainty representations–I: Generalized p-boxes,” *International Journal of Approximate Reasoning*, vol. 49, no. 3, pp. 649–663, 2008, doi: 10.1016/j.ijar.2008.07.003.

[10] R. C. Williamson and T. Downs, “Probabilistic arithmetic. I. Numerical methods for calculating convolutions and dependency bounds,” *International Journal of Approximate Reasoning*, vol. 4, no. 2, pp. 89–158, 1990, doi: 10.1016/0888-613x(90)90022-t.

[11] S. Ferson, V. Kreinovich, L. Ginzburg, D. S. Myers, and K. Sentz, *Constructing probability boxes and Dempster-Shafer structures*, no. 4015. Sandia National Laboratories, 2003. doi: 10.2172/809606.

[12] N. Gray, S. Ferson, M. De Angelis, A. Gray, and F. B. de Oliveira, “Probability bounds analysis for Python,” *Software Impacts*, vol. 12, p. 100246, 2022, doi: https://doi.org/10.1016/j.simpa.2022.100246.

[13] S. Ferson, V. Kreinovich, J. Hajagos, W. L. Oberkampf, and L. Ginzburg, “Experimental uncertainty estimation and statistics for data having interval uncertainty.,” 2007. doi: 10.2172/910198.

[14] M. de Angelis, “intervals.” [Online]. Available: https://doi.org/10.5281/zenodo.6205624

[15] K. Tretiak and S. Ferson, “Should data ever be thrown away? Pooling interval-censored data sets with different precision,” *International Journal of Approximate Reasoning*, vol. 156, pp. 114–133, 2023, doi: 10.1016/j.ijar.2023.02.007.

[16] S. Ferson *et al.*, “Natural language of uncertainty: numeric hedge words,” *International Journal of Approximate Reasoning*, vol. 57, pp. 19–39, 2015, doi: 10.1016/j.ijar.2014.11.003.

[17] C. Donnelly and P. Embrechts, “The devil is in the tails: actuarial mathematics and the subprime mortgage crisis,” *ASTIN Bulletin: The Journal of the IAA*, vol. 40, no. 1, pp. 1–33, 2010, doi: 10.2143/ast.40.1.2049222.

[18] W. L. Oberkampf *et al.*, “Dependence in probabilistic modeling, Dempster-Shafer theory, and probability bounds analysis.,” 2004. doi: 10.2172/919189.

[19] L. G. Crespo and S. P. Kenny, “The NASA langley challenge on optimization under uncertainty,” *Mechanical Systems and Signal Processing*, vol. 152, p. 107405, 2021, doi: 10.1016/j.ymssp.2020.107405.

[20] H. Zhang, R. L. Mullen, and R. L. Muhanna, “Interval Monte Carlo methods for structural reliability,” *Structural Safety*, vol. 32, no. 3, pp. 183–190, 2010, doi: 10.1016/j.strusafe.2010.01.001.