Probability-box propagation with interval prediction models: a reproducible Python workflow

Juntao Hu[1], Yu Chen[2], and Edoardo Patelli[1]
[1] *Department of Civil and Environmental Engineering, University of Strathclyde, Glasgow, UK*
juntao.hu@strath.ac.uk, edoardo.patelli@strath.ac.uk
[2] *Mechanical and Aerospace Engineering, University of Liverpool, Liverpool, UK*
yu.chen2@liverpool.ac.uk

Abstract. Probability boxes, or p-boxes, provide a useful representation of uncertainty when a precise probability distribution cannot be justified from the available information. Although p-box propagation is well established theoretically, its practical implementation remains challenging for users who require transparent and reproducible computational workflows. This paper presents a tutorial-style Python workflow for constructing and propagating p-boxes and for approximating propagated response bounds using Interval Prediction Models via a newly developed Python package `PyUncertainNumber`. The aim is not to introduce a new propagation theory, but to clarify how p-box inputs can be defined, propagated through nonlinear models, extracted as probability-level-dependent response intervals, and represented by a compact interval model. The workflow is first demonstrated using the Rosenbrock function, which provides a simple nonlinear benchmark. It is then extended to an excavation-induced tunnel displacement problem, where a first-level Interval Prediction Model is used as a surrogate of numerical simulation data and soil-parameter uncertainty is propagated to tunnel displacement predictions. The results show how p-box propagation and interval modelling can be combined into a reproducible workflow for conservative engineering response assessment. The paper is intended as a practical guide for researchers and engineers implementing imprecise-probability propagation in software-based uncertainty analysis.

Keywords: Probability box; Uncertain number; Interval Prediction Model; Tutorial

1. Introduction

Engineering systems are commonly affected by multiple sources of uncertainty, including natural variability, limited experimental data, measurement errors, model-form simplifications, and incomplete knowledge of the physical system. In reliability analysis and risk-informed decision-making, these uncertainties must be propagated through computational models in order to assess the possible range of system responses. This task becomes particularly important when the predicted response is used for safety evaluation, design verification, or failure probability estimation, see e.g. (Nuttall et al., 2025). In many engineering applications, uncertainty is traditionally represented using precise probabilistic models. However, sufficient data are not always available to justify a precise distributional assumption. In such cases, assigning a specific probability distribution may lead to overconfident predictions.

To address this issue, imprecise probability models have been developed to represent uncertainty under limited or incomplete information (Faes et al., 2021). Probability boxes, or P-boxes, provide a flexible representation of imprecise probabilistic information. Instead of defining a single cumulative distribution function, a P-box describes a family of admissible distributions bounded by lower and upper cumulative distribution functions. This allows aleatory variability and epistemic uncertainty to be considered simultaneously. P-boxes have therefore been widely used in engineering analysis when probability distributions are only partially known, when data are sparse, or when multiple sources of uncertain information must be combined (Lye et al., 2022; Zhang et al., 2023; Chen and Ferson, 2025).

The propagation of P-boxes through computational models remains a challenging task. Compared with conventional probabilistic uncertainty propagation, P-box propagation often requires the evaluation of response bounds over a family of possible input distributions. Direct approaches, such as double-loop sampling, interval Monte Carlo, or repeated optimisation, can become computationally demanding, especially when the model is nonlinear, high-dimensional, or expensive to evaluate. This difficulty is particularly relevant for engineering problems involving high-fidelity numerical models, where each model evaluation may require considerable computational resources.

Several efficient methods have been proposed to reduce the computational cost of P-box propagation (Patelli et al., 2015). Sparse polynomial chaos expansions have been used to propagate P-box uncertainty through complex computational models with a reduced number of model evaluations (Schöbi and Sudret, 2017). Sparse-decomposition-based polynomial chaos expansions have also been developed for parameterized P-boxes, aiming to obtain probability bounds and statistical moment bounds more efficiently (Liu et al., 2020). For dependent uncertain inputs, copula-based methods have been introduced to propagate correlated parametric P-boxes in structural models (Liu et al., 2021). More recently, meta-model-based strategies, such as active-learning Kriging, have been proposed for uncertainty propagation involving non-parameterized P-boxes (Zhang et al., 2023). These studies demonstrate the importance of surrogate-assisted P-box propagation, but most existing approaches are primarily designed to approximate response functions or response distributions rather than to construct conservative interval response envelopes directly.

Interval Predictor Models provide a data-driven framework for constructing response bounds from available data (Sadeghi et al., 2018). Unlike conventional regression or surrogate models that aim to approximate a single deterministic response or a conditional mean, an Interval Predictor Model seeks to enclose observed responses within lower and upper prediction functions. The original formulation of interval predictor models with linear parameter dependency provides a basis for constructing bounded response predictions under uncertainty (Crespo et al., 2016). This interval-valued structure makes IPMs particularly attractive for engineering applications where conservative lower and upper response envelopes are more informative than a single best-estimate prediction. The use of Interval Predictor Models for P-box propagation has been investigated as a robust strategy for propagating imprecise probabilistic information through computational models (Sadeghi et al., 2020). In this setting, the IPM acts as an interval-based surrogate that maps uncertain inputs to lower and upper response bounds. Compared with point-valued surrogate models, this is particularly useful when the objective is not only to approximate the model response, but also to retain conservative response bounds throughout the uncertainty propagation process. However, further development is required to extend this idea to multidimensional uncertain input spaces and to clarify its role as an efficient surrogate framework for engineering applications involving limited data and expensive numerical simulations.

The contribution of this paper is therefore practical rather than theoretical. It provides: (i) a reproducible Python workflow for constructing parametric p-boxes and propagating them through nonlinear models; (ii) a clear distinction between a first-level IPM used as a surrogate of an engineering model and a second-layer IPM used to approximate a propagated output p-box; (iii) a tutorial benchmark based on the Rosenbrock function; and (iv) an engineering demonstration involving excavation-induced tunnel displacement, where uncertainty in soil parameters is propagated through an IPM surrogate. The paper is intended to help users implement p-box propagation in a transparent and auditable way, rather than to propose a new propagation algorithm.

2. P-box propagation and IPM approximation workflow

This section defines the methodological and software workflow used in the remainder of the paper. The aim is not to provide a complete theoretical review of probability boxes or Interval Prediction Models, but to introduce the concepts required to implement the proposed tutorial. The workflow combines three components: p-box representation of imprecise probabilistic inputs, propagation through a model or surrogate, and interval approximation of the propagated output p-box.

A central distinction in this paper is between two different uses of Interval Prediction Models. A first-level IPM may be used as a surrogate of an expensive engineering model. In contrast, a second-layer IPM is used after p-box propagation to approximate the extracted output p-box bounds as functions of probability level. These two uses are conceptually different and should not be interpreted as the same modelling task.

2.1. Probability boxes

A probability box, or p-box, represents uncertainty by bounding a family of cumulative distribution functions. Instead of assuming one precise cumulative distribution function $F_X(x)$, a p-box is defined by lower and upper cumulative probability bounds:

$$\underline{F}_X(x) \leq F_X(x) \leq \overline{F}_X(x), \tag{1}$$

where $\underline{F}_X(x)$ and $\overline{F}_X(x)$ denote the lower and upper cumulative distribution bounds, respectively. This representation is useful when the available information is insufficient to identify a unique probability distribution, but sufficient to constrain a family of admissible distributions.

In this paper, p-boxes are mainly used to represent uncertain input variables whose distributional parameters are not precisely known. For example, a normally distributed variable with uncertain mean and uncertain standard deviation can be expressed as

$$X \sim \mathcal{N}(\mu, \sigma^2), \qquad \mu \in [\underline{\mu}, \overline{\mu}], \qquad \sigma \in [\underline{\sigma}, \overline{\sigma}]. \tag{2}$$

and the corresponding p-box encloses the cumulative distribution functions associated with this distribution family.

2.1.1. *P-box propagation*

The propagation stage takes uncertain inputs represented as p-boxes

$$\mathbf{X}_{\text{pbox}} = \left[X_{1,\text{pbox}}, X_{2,\text{pbox}}, \ldots, X_{d,\text{pbox}}\right]. \tag{3}$$

and propagates them through model

$$\mathbf{X}_{\text{pbox}} \to g(\mathbf{X}) \to Y_{\text{pbox}}. \tag{4}$$

The output Y_{pbox} describes the imprecise probabilistic uncertainty in the model response induced by the input p-boxes. In practice, the propagation of p-boxes is often perfomed using the slicing methods that propagates a p-box by discretising its probability axis into a finite number of probability levels. For each probability level α_i, the input p-box is represented by an interval of admissible quantile values,

$$X_{\alpha_i} = \left[\underline{x}(\alpha_i), \overline{x}(\alpha_i)\right], \qquad i = 1, \ldots, M. \tag{5}$$

For multiple uncertain inputs, each input p-box is sliced in the same way. At probability level α_i, the corresponding uncertain input domain is therefore

$$\mathcal{X}_{\alpha_i} = X_{1,\alpha_i} \times X_{2,\alpha_i} \times \cdots \times X_{d,\alpha_i}. \tag{6}$$

The response interval associated with this probability level is obtained by bounding the model response over the sliced input domain:

$$\underline{y}(\alpha_i) = \min_{\mathbf{x} \in \mathcal{X}_{\alpha_i}} g(\mathbf{x}), \qquad \overline{y}(\alpha_i) = \max_{\mathbf{x} \in \mathcal{X}_{\alpha_i}} g(\mathbf{x}). \tag{7}$$

Repeating this operation for all probability levels gives a discrete representation of the propagated output p-box:

$$Y_{\text{pbox}} = \left\{\alpha_i, \underline{y}(\alpha_i), \overline{y}(\alpha_i)\right\}_{i=1}^{M}. \tag{8}$$

Thus, the slicing procedure converts p-box propagation into a sequence of interval propagation problems. The probability levels preserve the probabilistic ordering of the input uncertainty, while the interval bounds at each level account for the epistemic uncertainty associated with the admissible family of input distributions. For nonlinear or non-monotonic models, the lower and upper response bounds should be obtained using an interval-bounding or optimisation strategy rather than by evaluating only the interval endpoints.

2.2. Interval Prediction Models

An Interval Prediction Model is a data-driven model that constructs lower and upper response bounds from input–output data. IPMs are used because their interval-valued structure is consistent with the objective of conservative response prediction under uncertainty. However, it is important to distinguish between two different use of IPM in the workflow, see Table I. The first-level IPM is a surrogate of a physical, numerical, or mathematical model. The second-layer IPM is a compact approximation of the p-box bounds obtained after uncertainty propagation through a model.

Table I. IPM roles in the proposed workflow.

Component	Input-Output	Purpose
First-level IPM	Measurements or model inputs-outputs	Regression analysis or surrogate of an expensive computational model
Second-layer IPM	Probability level α and corresponding response	Compact approximation of the propagated output p-box

2.2.1. *First-level IPM construction*

A first-level IPM is constructed from input–output data obtained from experimental values or from a computational model that maps input variables to a response quantity. The input samples may be generated using a deterministic design of experiments, random sampling, Latin Hypercube Sampling, or user-defined engineering scenariosn obtained a dataset:

$$\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^{N}\,, \tag{9}$$

. An IPM learns lower and upper prediction functions:

$$\mathrm{IPM}_1(\mathbf{x}) = [y_L(\mathbf{x}), y_R(\mathbf{x})]\,, \tag{10}$$

where $y_L(\mathbf{x})$ and $y_R(\mathbf{x})$ denote the lower and upper bounds of the interval prediction. A fitted IPM is normally required to enclose the training responses,

$$y_L(\mathbf{x}_i) \leq y_i \leq y_R(\mathbf{x}_i), \qquad i = 1, \ldots, N, \tag{11}$$

while keeping the prediction interval as compact as possible.

Once trained, the first-level IPM can be evaluated more efficiently than the original model and can therefore be used during the p-box propagation stage.

2.2.2. *Second-layer IPM approximation of the output p-box*

The second-layer IPM is obtained from the propagation of the uncertainty in a form of p-box through a computational model (or surrogate). An interval-valued mapping from probability level to response bounds:

$$\alpha_i \to [Y_L(\alpha_i), Y_R(\alpha_i)]\,. \tag{12}$$

and a IPM is then fitted to provide a bounded functional approximation of the p-box envelope:

$$\mathrm{IPM}_2 : \alpha \to \left[\widehat{Y}_L(\alpha), \widehat{Y}_R(\alpha)\right]. \tag{13}$$

The second-layer IPM is not the only possible approximation strategy. Alternative approaches, such as interpolation, spline approximation, monotone regression, or direct storage of the extracted p-box bounds, could also be used. The IPM is adopted because it provides an interval-valued representation consistent with the broader interval-modelling framework used in this paper.

2.3. Software components

Two uncertainty-related Python libraries are used. `pyuncertainnumber` (Chen and Ferson, 2025) is used to construct and propagate p-boxes. `PyIPM` (Sadeghi, 2020) is used to construct the first-level and second-layer Interval Prediction Models. `PyIPM` is a Python implementation of the Interval Predictor Model toolbox originally developed in the OpenCOSSAN uncertainty quantification framework (Patelli et al., 2017). Alternative implementations also exist, including OpenCOSSAN in MATLAB (Patelli et al., 2014) and ProbabilityBoundsAnalysis.jl in Julia (Gray et al., 2021).

The p-box and IPM components are connected through data transformations rather than treated as a single algorithm. First, uncertain inputs are defined either directly as p-boxes, as in the Rosenbrock tutorial, or through physical parameters, as in the excavation-induced tunnel displacement demonstration. The model response is then propagated through either an explicit mathematical function, a numerical model, or a first-level IPM surrogate. The resulting probability-level-dependent response bounds are finally reformulated as an interval dataset for the second-layer IPM approximation.

3. Tutorial example: Rosenbrock function

This section provides an executable tutorial example to demonstrate how input p-boxes are constructed, propagated through a nonlinear model, extracted as probability-level-dependent response bounds, and approximated using a second-layer IPM. The Rosenbrock function is used because it is simple to evaluate while still sufficiently nonlinear to produce an informative output p-box. Since the Rosenbrock function can be evaluated directly, no first-level IPM is required in this tutorial example.

3.1. Model definition

The two-dimensional Rosenbrock function is defined as

$$g(x_1, x_2) = (1 - x_1)^2 + 100(x_2 - x_1^2)^2. \tag{14}$$

It is treated as a generic black-box model with uncertain input variables X_1 and X_2. Listing 1 gives the Python implementation used in the propagation.

```python
def rosenbrock_model(*args):
    """ Rosenbrock function.
    It can accept either rosenbrock_model([x1, x2]) or rosenbrock_model(x1, x2). """
    if len(args) == 1:
        x1, x2 = args[0][0], args[0][1]
    elif len(args) == 2:
        x1, x2 = args
    else:
        raise ValueError("Rosenbrock model expects one or two input variables.")

    return (1 - x1)**2 + 100 * (x2 - x1**2)**2
```

Listing 1: Definition of the Rosenbrock benchmark function.

3.2. Input p-box construction

The two input variables are represented as parametric p-boxes. Each variable is assumed to follow a normal distribution with uncertain mean and uncertain standard deviation:

$$X_1 \sim \mathcal{N}(\mu_1, \sigma_1^2), \qquad \mu_1 \in [-1.2, -0.8], \qquad \sigma_1 \in [0.10, 0.20], \tag{15}$$

and

$$X_2 \sim \mathcal{N}(\mu_2, \sigma_2^2), \qquad \mu_2 \in [0.8, 1.2], \qquad \sigma_2 \in [0.10, 0.20]. \tag{16}$$

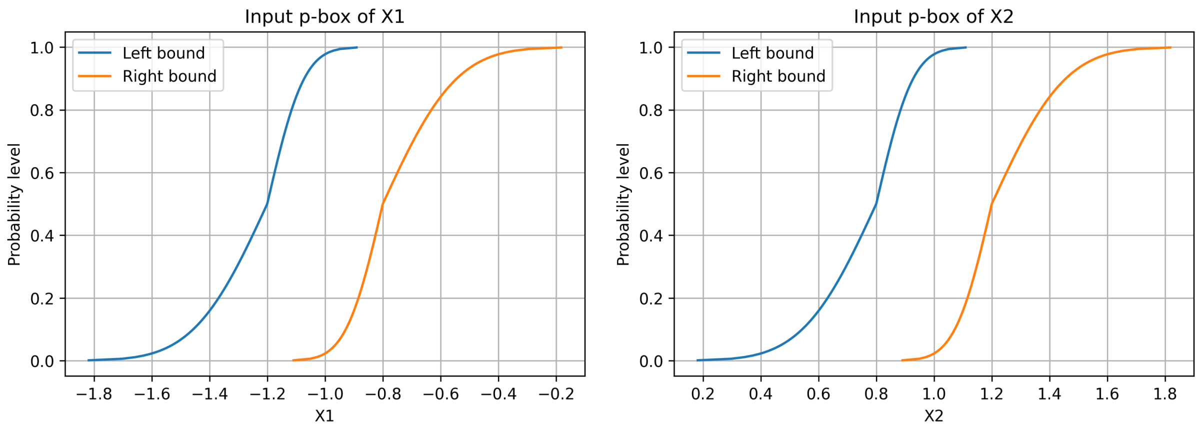
The corresponding p-boxes are constructed using `PyUncertainNumber`, as shown in Listing 2.

```python
from pyuncertainnumber import pba
from pyuncertainnumber import slicing

# Construct normal p-boxes with interval-valued parameters
x1_pbox = pba.normal([-1.2, -0.8], [0.10, 0.20])
x2_pbox = pba.normal([0.8, 1.2], [0.10, 0.20])
```

Listing 2: Construction of input p-boxes using PyUncertainNumber.

The resulting input p-boxes are shown in Figure 1.



(a) X_1 (b) X_2

Figure 1. Input p-boxes for the Rosenbrock example.

3.3. Direct p-box propagation

The input p-boxes are propagated through the Rosenbrock function using the slicing-based procedure introduced in Section 2.1.1. The corresponding implementation is shown in Listing 3, and the directly propagated output p-box is shown in Figure 2.

```python
# Run slicing-based p-box propagation
y_pbox = slicing(vars=[x1_pbox, x2_pbox],func=rosenbrock_model,
    interval_strategy="direct", n_slices=20, outer_discretisation=True)
```

Listing 3: Propagation of input p-boxes through the Rosenbrock function.

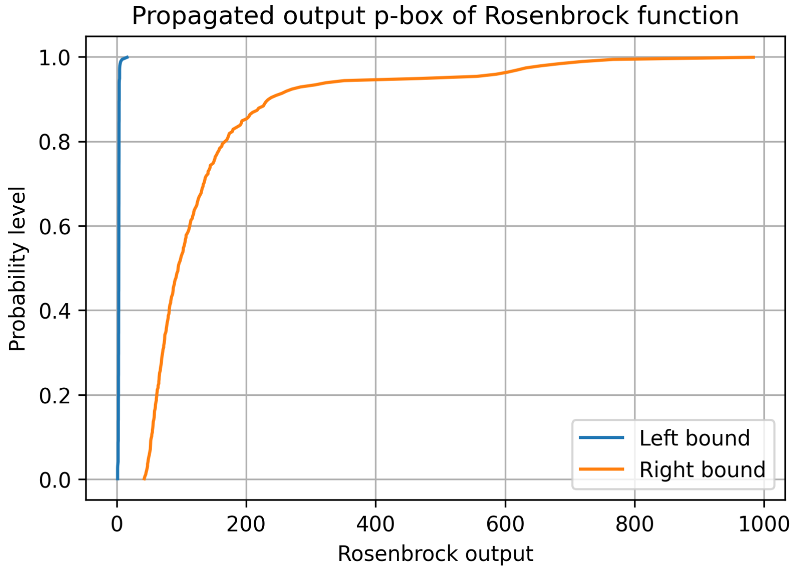


Figure 2. Output p-box of the Rosenbrock response propagated using `PyUncertainNumber`.

The spread between the left and right cumulative bounds reflects the epistemic uncertainty associated with the interval-valued distribution parameters of the inputs. The nonlinear form of the Rosenbrock function also induces an asymmetric response structure, which is visible in the large range of the propagated output bounds.

3.4. Extraction of output p-box bounds

Following Section 2.2.2, the propagated output p-box is extracted into arrays of probability levels and corresponding left and right response bounds. These arrays form the training data for the second-layer IPM. The extraction uses only public attributes of the `Pbox` object, as shown in Listing 3.

```python
def extract_pbox_bounds(pbox):
    """Extract probability levels and p-box bounds using public attributes.
    No private attributes such as _construct, _left, or _right are used."""
    alpha = np.asarray(pbox.p_values).reshape(-1, 1)
    y_left = np.asarray(pbox.left)
    y_right = np.asarray(pbox.right)
    return alpha, y_left, y_right

alpha, y_left, y_right = extract_pbox_bounds(y_pbox)
```

Listing 3: Extraction of output p-box probability levels and response bounds.

3.5. Second-layer IPM approximation

The second-layer IPM follows the mapping defined in Section 2.2.2 and it is used fit the Rosenbrock output p-box bounds. The Rosenbrock's response has a long right tail, therefore a log scale of the output space is used to fit the IPM. Since the logarithmic transformation is monotonic for positive responses, the ordering of the lower and upper response bounds is preserved after back-transformation. The implementation is shown in Listing 4.

```
import PyIPM
# Avoid log(0) before applying the logarithmic transformation
eps = 1e-12
y_left_safe = np.maximum(y_left, eps)
y_right_safe = np.maximum(y_right, eps)

# Stack the lower and upper propagated bounds as training responses
alpha_train = np.vstack([alpha, alpha])
log_y_train = np.concatenate([np.log(y_left_safe),np.log(y_right_safe)])

# Fit a quadratic second-layer IPM
model_ipm2 = PyIPM.IPM(polynomial_degree=2)
model_ipm2.fit(alpha_train, log_y_train)

# Predict second-layer IPM bounds at the extracted probability levels
log_upper_pred, log_lower_pred = model_ipm2.predict(alpha)
log_upper_pred = np.asarray(log_upper_pred).ravel()
log_lower_pred = np.asarray(log_lower_pred).ravel()

# Transform the fitted bounds back to the original response scale
y_upper_ipm = np.exp(log_upper_pred)
y_lower_ipm = np.exp(log_lower_pred)
```

Listing 4: Second-layer IPM approximation of the extracted output p-box using PyIPM.

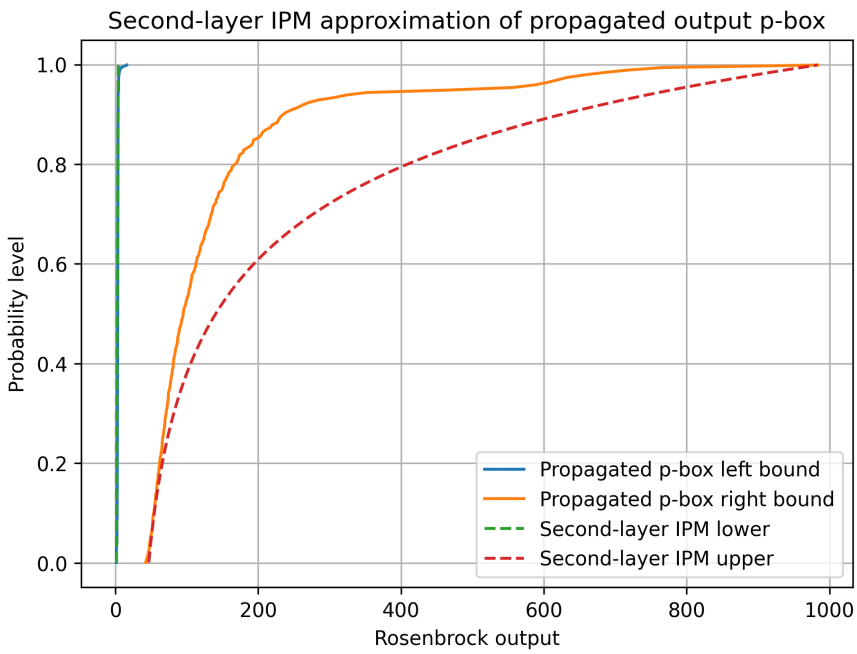


Figure 3. Second-layer IPM approximation of the Rosenbrock output p-box.

In this implementation, the extracted lower and upper response bounds are stacked into a single training set. The same probability levels are used for both sets of response values. The fitted bounds are then transformed back to the original response scale for visualisation. The resulting second-layer IPM approximation is shown in Figure 3.

The second-layer IPM should not be interpreted as an exact reconstruction of the propagated p-box. It is a compact interval approximation fitted to the extracted probability-level-dependent bounds. In this tutorial, a quadratic polynomial basis is deliberately used to keep the implementation simple and interpretable. The purpose is to demonstrate the computational connection between p-box propagation and interval approximation, rather than to optimise the approximation accuracy. If higher approximation accuracy were required, a higher-order basis or a more flexible approximation scheme could be used.

4. Engineering demonstration: excavation-induced tunnel displacement

This section demonstrates how the Rosenbrock tutorial workflow can be transferred to an engineering problem involving excavation-induced tunnel displacement. The purpose is not to present a complete site-specific reliability assessment, but to show how p-box propagation can be coupled with a first-level IPM surrogate trained from numerical simulation data. The excavation model, simulation dataset, and calibration procedure are reported in detail in the submitted journal paper (Hu et al., 2026). Therefore, this section focuses only on using the trained excavation IPM surrogate within the p-box propagation workflow.

The demonstration considers the horizontal displacement of an existing tunnel affected by a nearby excavation, a common urban geotechnical problem where ground movements may affect adjacent underground infrastructure.

4.1. First-level excavation IPM surrogate

The first-level IPM surrogate is trained from an excavation simulation dataset. The input vector is defined as

$$\mathbf{x} = \left[B, EI_{\text{ratio}}, \frac{s_u}{\sigma'_v}, \frac{E_{ur}}{\sigma'_v}, D_h, \frac{D_v}{H}\right], \tag{17}$$

where B is the excavation width, EI_{ratio} is the retaining system stiffness ratio, s_u/σ'_v is the undrained shear strength ratio, E_{ur}/σ'_v is the unloading stiffness ratio, D_h is the horizontal distance between the excavation and the tunnel, and D_v/H is the normalised tunnel depth. The response variable considered in this demonstration is the horizontal tunnel displacement u_x.

The first-level IPM maps the excavation input vector to lower and upper displacement predictions:

$$\text{IPM}_1(\mathbf{x}) = [u_{x,L}(\mathbf{x}), u_{x,R}(\mathbf{x})]. \tag{18}$$

In the present demonstration, the upper bound $u_{x,R}(\mathbf{x})$ is used as the conservative response for the subsequent p-box propagation:

$$U_x = u_{x,R}(\mathbf{x}). \tag{19}$$

This means that the propagation is performed through the upper envelope of the surrogate, rather than through a single deterministic best-estimate model.

The training performance of the first-level IPM surrogate is summarised in Table II. The reported lower-bound and upper-bound violations are close to numerical precision, indicating that the fitted IPM encloses the training responses.

Table II. Summary of the first-level IPM training results for horizontal tunnel displacement.

Item	Value
Number of training samples	1000
Number of input variables	6
Output variable	Horizontal displacement, u_x
Maximum lower-bound violation	2.65×10^{-12}
Maximum upper-bound violation	1.46×10^{-12}
Average interval width	0.6675
Minimum interval width	0.0507
Maximum interval width	3.2939

Table II confirms that the first-level IPM provides an enclosing surrogate over the training dataset. However, this training enclosure should not be interpreted as a complete validation of predictive accuracy outside the sampled domain. The surrogate is used here only within the range represented by the simulation data.

4.2. Soil-parameter uncertainty model

The input uncertainty is introduced through physically meaningful soil parameters rather than by directly prescribing uncertainty on the surrogate descriptors s_u/σ'_v and E_{ur}/σ'_v. Specifically, uncertainty is assigned to cohesion c, friction angle ϕ, and a stiffness multiplier λ. These parameters are then transformed into the strength and stiffness descriptors required by the first-level IPM.

The undrained shear strength ratio is computed as

$$\frac{s_u}{\sigma'_v} = \frac{c\cos\phi}{\sigma'_v} + \frac{1}{2}(1 + K_0)\sin\phi, \tag{20}$$

where σ'_v is the effective vertical stress and K_0 is the earth pressure coefficient at rest. The unloading stiffness ratio is then obtained from

$$\frac{E_{ur}}{\sigma'_v} = \lambda\frac{s_u}{\sigma'_v}, \tag{21}$$

where λ is an uncertain multiplicative coefficient. This transformation preserves the dependency between the strength and stiffness descriptors and avoids physically inconsistent strength–stiffness combinations. The remaining engineering variables are fixed at representative values to isolate the effect of soil-parameter uncertainty.

4.3. P-box propagation through the excavation surrogate

The p-box propagation is performed through the conservative upper response of the first-level IPM surrogate. At each probability slice, the uncertain soil parameters define admissible intervals for c, ϕ, and λ. These intervals are transformed through Equations (20) and (21) to obtain corresponding intervals for s_u/σ'_v and E_{ur}/σ'_v. The resulting uncertain engineering descriptors are then propagated through the first-level IPM using the conservative response defined in Equation (19).

Repeating this process over all probability levels gives the output p-box of the conservative horizontal displacement response. The propagated output p-box is shown in Figure 4. The vertical dashed line indicates a 20 mm displacement threshold used here as an illustrative engineering control limit.

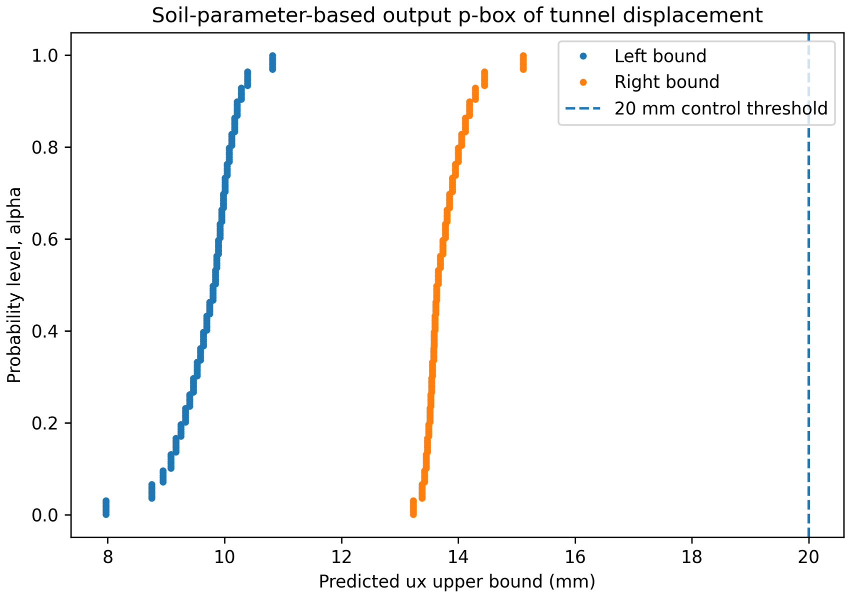


Figure 4. Soil-parameter-based output p-box of conservative horizontal tunnel displacement.

The output p-box provides more information than a deterministic displacement prediction. It shows the range of cumulative response behaviour induced by imprecise soil-parameter uncertainty. If the upper response bound remains below the 20 mm threshold over the relevant probability range, the scenario can be interpreted as conservatively acceptable under the specified uncertainty assumptions. If part of the upper response bound crosses the threshold, the result indicates that the uncertainty model admits soil conditions that may require additional design review, monitoring, or more detailed numerical analysis.

4.4. Second-layer approximation of the engineering output p-box

The same procedure introduced in Section 2.2.2 is used here to compute the bounds of the conservative horizontal displacement response obtained from the first-level excavation IPM surrogate.

The second-layer approximation is shown in Figure 5. The markers denote the propagated p-box bounds, while the fitted curves provide a compact interval representation of the same result.

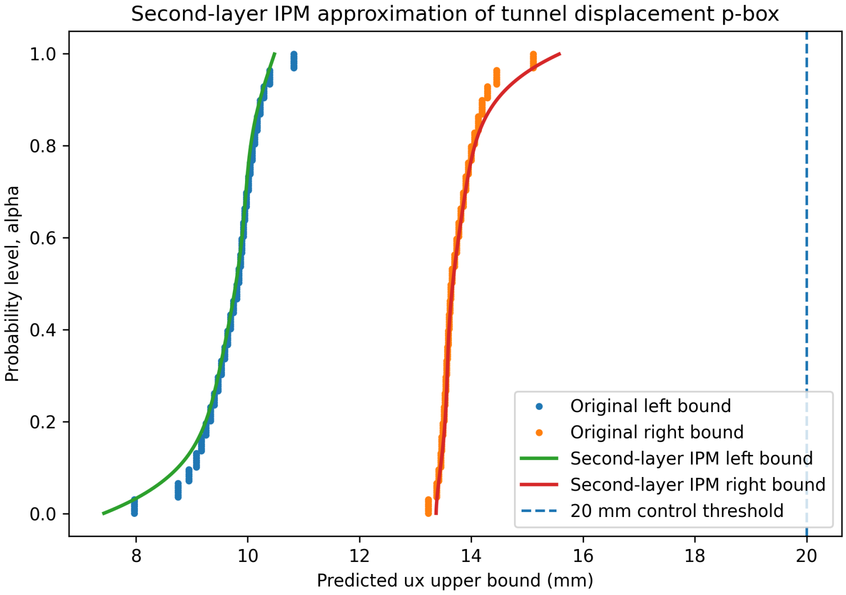


Figure 5. Second-layer IPM approximation of the soil-parameter-based output p-box.

This second-layer approximation is optional but useful when a compact functional representation of the propagated p-box is needed for storage, comparison, or subsequent calculations. However, it should be interpreted as an approximation of the already propagated p-box, not as a replacement for the first-level excavation surrogate or the original numerical simulation model.

4.5. Discussion and limitations of the engineering demonstration

The engineering demonstration shows how the tutorial workflow can be transferred from an analytical benchmark to a surrogate-based geotechnical problem. The uncertainty is introduced through physically interpretable soil parameters, while the expensive model response is replaced by an interval-valued surrogate. This makes it possible to combine p-box propagation with conservative response prediction in a computationally efficient way.

The demonstration also highlights the importance of preserving physical dependencies between derived inputs. In the present case, the strength and stiffness descriptors are both computed from the same underlying soil parameters. This avoids the physically inconsistent combinations that could arise if the two descriptors were treated as independent uncertain inputs.

Several limitations should be noted. First, the example is intended as a demonstration of the workflow, not as a complete site-specific reliability analysis. Only selected soil parameters are treated as uncertain, while other engineering variables are fixed at representative values. Second, the first-level IPM is used within the range of the simulation data from which it was trained. Its reliability outside this range is not assessed here. Third, the training enclosure reported in Table II demonstrates consistency with the training data, but it does not replace independent validation. Finally, the 20 mm threshold is used as an illustrative control limit; a real assessment would require a project-specific serviceability or damage criterion.

Future engineering applications should therefore include a broader uncertainty model, explicit treatment of dependency between geotechnical parameters, validation of the first-level IPM against independent simulations, and sensitivity analysis of the p-box propagation settings. Despite these limitations, the example demonstrates the practical role of the proposed workflow as a transparent bridge between imprecise soil-parameter uncertainty and conservative displacement assessment.

5. Conclusions

This paper presented a reproducible workflow for propagating probability boxes using a Python library called `pyuncertainnumber` and approximating the propagated output bounds using Interval Prediction Models. The contribution is practical rather than theoretical: the paper clarifies how p-box inputs can be constructed, propagated, extracted as probability-level-dependent response intervals, and represented by a compact second-layer IPM.

The Rosenbrock example showed the complete workflow in a controlled nonlinear setting. Two normal p-box inputs with interval-valued means and standard deviations were propagated through the Rosenbrock function using slicing-based propagation. The resulting output p-box was then extracted as pairs of lower and upper response bounds at discrete probability levels. A quadratic second-layer IPM, fitted in logarithmic response space, provided a compact approximation of this propagated p-box.

The excavation-induced tunnel displacement example showed how the same workflow can be used when the original model is replaced by a first-level IPM surrogate. In this case, uncertainty was introduced through cohesion, friction angle, and a stiffness multiplier, then transformed into the strength and stiffness descriptors required by the excavation surrogate. This avoided assigning independent uncertainty directly to derived variables and preserved a clearer physical link between soil uncertainty and tunnel displacement.

The main technical distinction is that the two IPMs have different roles. The first-level IPM is a surrogate of the engineering model. The second-layer IPM is not a model surrogate; it is an approximation of the propagated output p-box. This distinction is essential for interpreting the workflow correctly.

The results indicate that the proposed workflow is useful for tutorial and engineering-screening purposes, especially when users need an auditable connection between imprecise input distributions, p-box propagation, and conservative response bounds. The results indicate that the proposed workflow is useful for tutorial and engineering-screening purposes, especially when users need an auditable connection between imprecise input distributions, p-box propagation, and conservative response bounds. Future work should focus on broader uncertainty modelling, independent validation of the first-level IPM, and sensitivity analysis of the propagation settings.

References

Chen, Y. and S. Ferson: 2025, 'Imprecise uncertainty management with uncertain numbers to facilitate trustworthy computations'. In: *SciPy Proceedings*.

Crespo, L. G., S. P. Kenny, and D. P. Giesy: 2016, 'Interval predictor models with a linear parameter dependency'. *Journal of Verification, Validation and Uncertainty Quantification* **1**(2), 021007.

Faes, M., M. Daub, S. Marelli, E. Patelli, and M. Beer: 2021, 'Engineering analysis with probability boxes: A review on computational methods'. *Structural Safety* **93**, 102092.

Gray, A., S. Ferson, and E. Patelli: 2021, 'ProbabilityBoundsAnalysis.jl: Arithmetic with sets of distributions'. *Proceedings of the JuliaCon Conferences*.

Hu, J., E. Patelli, Y. Chen, and M. De Angelis: 2026, 'A Robust Data-Driven Interval Prediction of Tunnel Displacement Induced by Urban Excavation'. Manuscript submitted for publication.

Liu, H., M. Chen, C. Du, J. Tang, C. Fu, and G. She: 2021, 'A copula-based uncertainty propagation method for structures with correlated parametric p-boxes'. *International Journal of Approximate Reasoning* **138**, 89–104.

Liu, H. B., C. Jiang, and Z. Xiao: 2020, 'Efficient uncertainty propagation for parameterized p-box using sparse-decomposition-based polynomial chaos expansion'. *Mechanical Systems and Signal Processing* **138**, 106589.

Lye, A., M. Kitahara, M. Broggi, and E. Patelli: 2022, 'Robust Optimization of a Dynamic Black-box System under Severe Uncertainty: A Distribution-Free Framework'. *Mechanical Systems and Signal Processing* **167**, 108522.

Nuttall, W., E. Patelli, E. Smith, D. Webbe-Wood, and S. Middleburgh: 2025, *Perspectives on Engineering Uncertainty: Civil Nuclear Energy Safety and Efficiency*. Springer Nature. Accepted: 2025-05-13T08:23:33Z.

Patelli, E., D. A. Alvarez, M. Broggi, and M. d. Angelis: 2015, 'Uncertainty management in multidisciplinary design of critical safety systems'. *Journal of Aerospace Information Systems* **12**(1), 140–169.

Patelli, E., M. Broggi, M. d. Angelis, and M. Beer: 2014, 'OpenCossan: An Efficient Open Tool for Dealing with Epistemic and Aleatory Uncertainties'. In: *Vulnerability, Uncertainty, and Risk*. pp. 2564–2573, American Society of Civil Engineers.

Patelli, E., M. Broggi, S. Tolo, and J. Sadeghi: 2017, 'Cossan Software: A Multidisciplinary and Collaborative Software for Uncertainty Quantification'. In: *Proceedings of the 2nd ECCOMAS Thematic Conference on Uncertainty Quantification in Computational Sciences and Engineering*. Rhodes Island, Greece.

Sadeghi, J., M. D. Angelis, and E. Patelli: 2018, 'Frequentist history matching with interval predictor models'. *Applied Mathematical Modelling* **61**, 29–48.

Sadeghi, J., M. De Angelis, and E. Patelli: 2020, 'Robust propagation of probability boxes by interval predictor models'. *Structural Safety* **82**, 101889.

Sadeghi, J. C.: 2020, 'PyIPM: Interval Predictor Models in Python'. `https://github.com/JCSadeghi/PyIPM`. Python package, version 2.0.

Schöbi, R. and B. Sudret: 2017, 'Uncertainty propagation of p-boxes using sparse polynomial chaos expansions'. *Journal of Computational Physics* **339**, 307–327.

Zhang, K., N. Chen, J. Liu, S. Yin, and M. Beer: 2023, 'An efficient meta-model-based method for uncertainty propagation problems involving non-parameterized probability-boxes'. *Reliability Engineering & System Safety* **238**, 109477.