Probabilistic verification of neural networks with sampling-based Probability Box propagation.

Marcel Chwiałkowski[1], Eric Goubault[2], and Sylvie Putot[2]

[1] École Polytechnique, marcel.chwialkowski@polytechnique.edu
[2] LIX, École Polytechnique, CNRS, Institut Polytechnique de Paris, 91120 Palaiseau, France lastname@lix.polytechnique.fr

Abstract. In probabilistic neural network verification, a well-chosen representation of input uncertainty ensures that theoretical analyses accurately reflect real input perturbations. A recent approach based on probability boxes (p-boxes) [9] is introduced in [10] and unifies set-based and probabilistic information on the inputs. The method allows for obtaining guaranteed probabilistic bounds for property satisfaction on feedforward ReLU networks. However, it suffers from conservatism due to employing set-based propagation methods.
In this work we investigate how to sample from p-boxes without loss of information. Based on that, we develop a sampling-based approach for propagating p-boxes through feedforward ReLU networks. We prove that with dense enough coverings of the input p-boxes, the propagated samples accurately represent the output uncertainty and provide error bounds. Additionally, we show how to create coverings for arbitrary p-boxes with various distributions. On the ACAS Xu benchmark we demonstrate that our approach is applicable in practice, both as a standalone verifier and as a way to partially assess the conservatism of the set-based approach of [10].

Keywords: Probability boxes · Neural network verification.

1 Introduction

Neural networks have been proven to be powerful tools in a wide range of applications, from image recognition to autonomous systems, making their reliability and correctness critical. Neural network verification focuses on ensuring safety of trained neural networks. Specifically, verification can consist of assessing the robustness of neural networks when faced with input uncertainties or adversarial attacks. The task of verifying a neural network can be approached deterministically or probabilistically. In the classical, deterministic approach, the goal is to determine if a safety property holds for a bounded input uncertainty or provide deterministic bounds for property violation (i.e. how much of a perturbation is guaranteed to not violate the property, or conversely). Examples of such approaches include the CROWN [28], FROWN [14], and CNN-Cert [4] frameworks. Approaches such as DeepZ [21], DeepPoly [22] and Verinet [12] propagate abstractions representing input uncertainty to verify if given properties hold.

On the other hand, probabilistic verification assumes random uncertainty at input, for example generated by random noise applied to the input, and aims to provide statistical estimates on output given the input uncertainty. Much less work has been done in this domain, but examples include [24,29] using it to assess network robustness and [2,1,27,18,13] probabilistically certifying correctness under adversarial attacks. Some works, including the PROVEN [25] framework also provide guaranteed statistical estimates, that is statements of form "the probability of X occurring is guaranteed to be between a and b". Probabilistic methods for networks with ReLU activations have also been developed in [17], [19] [7]. A notable family of methods that facilitate estimating statistics of outputs is covariance propagation, with examples in [26] and [16].

The works mentioned above operate either in a set-based or probabilistic input setting. However, as mentioned in [10], this does not accurately reflect reality, as sometimes an input can be represented with more than one probabilistic model, or a probabilistic model might have uncertain parameters. With those considerations, a recent approach introduced in [10] uses inputs described by imprecise probabilities [23,3] which unify both set-based and probabilistic inputs. This approach allows for taking into account both epistemic and aleatoric uncertainty on inputs. One realisation of imprecise probabilities is the concept of a probability box (p-box) [9] - a set of cumulative distribution functions (CDF), bounded by a lower and an upper boundary CDF. Representing input uncertainty with a p-box permits the input value to come from any of these distributions.

Propagating p-boxes has been extensively studied in the context of differential equations - for example for ODEs, as in [6] and PDEs, as in [11]. Moreover, P-boxes can be transformed into Dempster-Shafer Interval structures (DSI) [9], their discrete over-approximations, and algorithms for arithmetic operations on DSI allow for propagating DSI through neural networks, as done in [10] for feed-forward ReLU networks. However, this approach suffers from conservativeness, and the resulting DSI might be much less tight than in reality. To alleviate that, [10] introduces a new abstraction, Zonotopic Dempster-Schafer structures (DSZ), which generalise DSI and produce much tighter results when propagating through neural networks. The authors of [10] use their propagation algorithms for DSI and DSZ to provide guaranteed probabilistic bounds on the outputs of the neural network satisfying a safety property. While both the probabilistic bounds and resulting DSI/DSZ are robust, their quality, i.e. how tight they are with respect to reality, is difficult to fully assess.

In this work we develop and prove the validity of a systematic approach for propagating p-boxes through neural networks. Our method is based on sampling from dense-enough coverings of input p-boxes, passing the samples through the network and constructing output p-boxes or verifying safety properties. Dependent on parameters of our methods and the Lipschitz constant of the underlying neural network, we also provide accuracy bounds for our results. Our method is applicable also without knowledge of the Lipschitz constant or when the Lipschitz constant is large, but in these cases it lacks the accuracy bounds on the

results. In the case of neural networks with inputs represented by parametric p-boxes, a particular class of p-boxes where the distributions they contain are explicitly parameterised, we are able to implement and demonstrate our approach on the ACAS Xu benchmark. We compare these results to the output p-box estimates produced by the DSZ propagation algorithm from [10], although the latter is not restricted to the parametric case.

Additionally, we demonstrate ways of constructing coverings of parametric and non-parametric p-boxes and prove error bounds for our method - that is, we show that any output which our method did not account for is sufficiently close to an output that our method produced. However, we chose not to report on sampling non-parametric p-boxes as, for now, the methods with guarantees we presented are of exponential complexity in the size of the p-boxes.

1.1 Notation

Throughout the paper, we adhere to the following notations:

- $\mathbf{X}, \mathbf{Y}$, etc. denote multi-dimensional random variables, while X, Y, etc. denote one-dimensional random variables. Samples corresponding to these variables are denoted as $\mathbf{x}$ and x, respectively.
- For each random variable X, F_X denotes its cumulative distribution function (CDF), given by $F_X(x) = \mathbb{P}(X \leq x)$.

2 Problem Statement

We consider a ReLU feedforward neural network f with n independent inputs and m outputs. Values fed to f belong to probabilistic input sets:

Definition 1 (Probabilistic input set). *For two cumulative distribution functions $\underline{F}$ and $\overline{F}$, a probabilistic input set is $\mathcal{X} = \{\mathbf{X} : \underline{F}(\mathbf{x}) \leq F_X(x) \leq \overline{F}(\mathbf{x}), \forall \mathbf{x}\}$. A value belongs to a probabilistic input set if it is a sample from one of the distributions within it.*

As the inputs to f are independent, considering one n-dimensional probabilistic input set $\mathcal{X}$ is equivalent to considering n marginal input sets $\mathcal{X}_1, \ldots \mathcal{X}_n$.

Probabilistic output sets A probabilistic output set is defined as $\mathcal{Y} = \{\mathbf{Y} \coloneqq f(\mathbf{X}) | \mathbf{X} \in \mathcal{X}\}$, and can also be expressed in terms of some boundary CDFs $\underline{F}$ and $\overline{F}$. Marginal output sets $\mathcal{Y}_i$ for $i \in [m]$ are defined as $\mathcal{Y}_i = \{\mathbf{Y}[i] \coloneqq f(\mathbf{X})[i] | \mathbf{X} \in \mathcal{X}\}$. Given $\mathcal{Y}$, it is possible to derive $\mathcal{Y}_1, \ldots \mathcal{Y}_m$. However, since the outputs of f are not guaranteed to be independent, knowing $\mathcal{Y}_1 \ldots \mathcal{Y}_m$ alone is not sufficient to characterise $\mathcal{Y}$.

We solve the following problems:

P1 Find approximate marginal output sets $\hat{\mathcal{Y}}_1, \ldots \hat{\mathcal{Y}}_m$, such that for a given $\varepsilon > 0$, and $i \in [m]$, for any $Y \in \mathcal{Y}_i$ there exists $\hat{Y} \in \hat{\mathcal{Y}}_i$ s.t. $||F_Y - F_{\hat{Y}}||_1 \leq \varepsilon$,

P2 Approximate the probability bounds of an output vector $\mathbf{y}$ satisfying a linear safety property $A\mathbf{y} \leq v$.

The work of [10] solves variants of these problems using DSZ propagation - it finds an approximate output set $\hat{\mathcal{Y}}$, guaranteed to contain $\mathcal{Y}$. Similarly, it finds guaranteed probabilistic bounds of an output vector $\mathbf{y}$ satisfying a linear safety property.

Our contributions include a sampling-based method for solving these problems, which can be used either as a standalone verifier or to assess the conservativeness of the bounds derived by [10].

3 Preliminaries

Probabilistic input sets are represented by probability boxes.

Definition 2 (P-box [9]). *Let 2 cumulative distribution functions $\underline{F}, \overline{F}$ satisfy $\underline{F}(x) \leq \overline{F}(x)$ for every $x \in \mathbb{R}$. The* probability box*:*

$$\mathcal{X} = [\underline{F}, \overline{F}] := \left\{\underline{F} \leq F \leq \overline{F}\right\}$$

is the set of CDFs bounded below by $\underline{F}$ and above by $\overline{F}$. A random variable X belongs *to $\mathcal{X}$ (written $X \in \mathcal{X}$) if and only if $F_X \in \mathcal{X}$.*

Expressing uncertainty A p-box encodes a measurement that exhibits both aleatoric and epistemic uncertainty. The span of the p-box reflects epistemic uncertainty, while the shape of each admissible F represents aleatoric uncertainty.

Definition 3 (Parametric p-box). *Let $\{F_\theta\}_{\theta \in \Theta}$ be a distribution family with Θ_0 being the set of admissible parameters. Then $\mathcal{X} = \{F_\theta \mid \theta \in \Theta_0\}$ is a parametric p-box; for instance $\mathcal{N}\big(\mu \in [0,1], \sigma = 1\big)$ bounds every Gaussian with mean in $[0,1]$ and unit variance.*

A parametric p-box corresponds to a measurement with a parametrised epistemic uncertainty - for example, when the mean of the data is known to belong to an interval $[a, b]$, but the actual value is unknown.

3.1 Sampling from p-boxes

We present a method for sampling values from p-boxes that allows for constructing p-boxes from data. For a p-box $\mathcal{X}$:

1. **Distribution selection** - A distribution $X \in \mathcal{X}$ is chosen,
2. **Sampling from the distribution** - s samples are drawn from X,
3. **Propagation** - For each sample x, $f(x)$ is calculated and the empirical distribution of $f(X) \in f(\mathcal{X})$ is calculated.

Repeating the process with different distribution selections $X_1, X_2, \ldots$ yields outputs $f(X_1), f(X_2), \ldots$ which can approximate the shape of $f(\mathcal{X})$.

It is desirable to pick distributions $X_1, X_2, \ldots X_i$ that approximate $\mathcal{X}$ well in order to lose as little information conveyed by $\mathcal{X}$ as possible. This is formalised by the concept of covering:

Definition 4 (Covering). *Let $\mathcal{X}$ be a p-box and ε a positive real number. A subset $\hat{\mathcal{X}} \subseteq \mathcal{X}$ is an ε-covering of $\mathcal{X}$ if for any $X \in \mathcal{X}$ there exists $\hat{X} \in \hat{\mathcal{X}}$ such that $||F_X - F_{\hat{X}}||_1 \leq \varepsilon$.*

Given a covering $\hat{\mathcal{X}}$ of $\mathcal{X}$ we can rerun the sampling process, each time choosing a different distribution from $\hat{\mathcal{X}}$ to sample from distributions representing the overall shape of the input.

3.2 General sampling-based p-box propagation

Sampling from multivariate probabilistic input set is similar. We are interested specifically in finite coverings - Algorithm 1 defines a procedure that can be used to solve problems **P1** and **P2** given finite coverings of the input p-boxes $\hat{\mathcal{X}}_1, \ldots \hat{\mathcal{X}}_n$. The correctness of algorithm 1 and the ε constant of the coverings that it creates are specified in the next section.

Algorithm 1: Sampling-based p-box propagation

Input: finite coverings of the input p-boxes, $\hat{\mathcal{X}}_1, \ldots \hat{\mathcal{X}}_n$
Output: finite coverings of the output p-boxes $\hat{\mathcal{Y}}_1 \ldots \hat{\mathcal{Y}}_m$
foreach $\mathbf{X} \coloneqq (X_1, \ldots X_n)$ *s.t.* $X_1 \in \hat{\mathcal{X}}_1, \ldots X_n \in \hat{\mathcal{X}}_n$ **do**
    for $j \leftarrow 1$ **to** s **do**
        Draw $\mathbf{x_{i,j}} \sim \mathbf{X}_i$
        $\mathbf{y_{i,j}} \leftarrow f(\mathbf{x}_{i,j})$;
    $\hat{\mathbf{Y}}_i \leftarrow$ Empirical Distribution$(\mathbf{y}_{i,1}, \ldots \mathbf{y}_{i,s})$
    Calculate the marginals of $\hat{\mathbf{Y}}_i$, i.e. $\hat{Y}_i^1, \ldots \hat{Y}_i^m$
    for $j \leftarrow 1$ **to** m **do**
        $\hat{\mathcal{Y}}_j \leftarrow \hat{Y}_i^j$
return $\{\hat{\mathcal{Y}}_1, \ldots \hat{\mathcal{Y}}_m\}$

Complexity Algorithm 1 propagates $\prod_{i=1}^{n} |\hat{\mathcal{X}}_i|$ distributions through the network. When propagating a single distribution, s inputs vectors are passed forward and m marginal distributions are built from the outputs. We assume that passing forward an input vector takes constant time, thus the cost associated to sample propagation is $O(s)$. Building a single marginal output distribution takes $O(s \log s)$, as it requires sorting all samples with respect to one of the

coordinates. As m distributions need to be built, and each time the sorting is done over a different coordinate, building m marginal output distributions takes $O(ms \log s)$. Overall, propagating a single distributions costs $O(ms \log s)$, so the complexity of Algorithm 1 is $O(ms \log sE^n)$, where $E = \sup_{i \in [n]} |\hat{\mathcal{X}}_i|$.

4 Main theoretical result

We show that for Lipschitz continuous neural networks, Algorithm 1 yields sets of distributions $\hat{\mathcal{Y}}_1, \dots \hat{\mathcal{Y}}_m$ which form coverings of the marginal output sets $\mathcal{Y}_1, \dots \mathcal{Y}_m$ and we quantify the precision of these coverings.

Theorem 1. *Given a neural network f with n independent inputs, m outputs, an $||\cdot||_1$-norm Lipschitz constant of at most L and ε-coverings for each input, Algorithm 1 produces a $nL\varepsilon$-covering of each marginal output set.*

Below we introduce Wasserstein distance which allows for passing random variables through Lipschitz continuous functions:

Definition 5 (Wasserstein Distance). *Wasserstein-k distance for $k \in \mathbb{N}$ between two random variables $\mathbf{P}$ and $\mathbf{Q}$ is defined as:*

$$W_k(P, Q) = \inf_{\gamma \in \Pi(P,Q)} (\mathbb{E}_{(\mathbf{x},\mathbf{y})\sim\gamma}[d(\mathbf{x}, \mathbf{y})^k])^{\frac{1}{k}}$$

where d is a distance and Π is the set of all couplings between $\mathbf{P}$ and $\mathbf{Q}$. For one-dimensional distributions P and Q it holds that:

$$W_1(P, Q) = ||F_P - F_Q||_1$$

We state properties of the Wasserstein distance relevant to our problem, with proofs given in the appendix.

Lemma 1 (Subadditivity of Wasserstein distance). *For 2 n-dimensional random vectors $\mathbf{X} \coloneqq (X_1, X_2, \dots X_n)$ and $\mathbf{Y} \coloneqq (Y_1, Y_2, \dots Y_n)$ equipped with the $||\cdot||_1$ norm it holds that:*

$$W_1(\mathbf{X}, \mathbf{Y}) \leq W_1(X_1, Y_1) + W_1\left(\begin{pmatrix} X_2 \\ \vdots \\ X_n \end{pmatrix}, \begin{pmatrix} Y_2 \\ \vdots \\ Y_n \end{pmatrix}\right)$$

Lemma 2 (Projection of Wasserstein distance). *.For 2 n-dimensional random vectors $\mathbf{X} = (X_1, X_2 \dots X_n)$ and $\mathbf{Y} = (Y_1, Y_2 \dots Y_n)$ equipped with the $||\cdot||_1$ norm it holds that:*

$$W_1(X_1, Y_1) \leq W_1(\mathbf{X}, \mathbf{Y})$$

Lemma 3 (Passing Wasserstein-1 distance through Lipschitz functions). *For a Lipschitz function f with a $||\cdot||_1$-norm Lipschitz constant of at most L and 2 random vectors $\mathbf{X}$ and $\mathbf{Y}$ it holds that:*

$$W_1(f(\mathbf{X}), f(\mathbf{Y})) \leq LW_1(\mathbf{X}, \mathbf{Y})$$

We prove our main result, Theorem 1:

Proof. Let $\mathcal{X}_1, \mathcal{X}_2 \dots \mathcal{X}_n$ denote the input p-boxes and $\hat{\mathcal{X}_1}, \hat{\mathcal{X}_2} \dots \hat{\mathcal{X}_n}$ their ε-coverings. Consider n random variables $X_1, \dots X_n$ with their CDFs $F_1, \dots F_n$ in the respective input p-boxes. By definition of a covering, there exist $\hat{X}_1, \hat{X}_2 \dots \hat{X}_n$ with CDFs $\hat{F}_1 \in \hat{\mathcal{X}_1}, \hat{F}_2 \in \hat{\mathcal{X}_2} \dots \hat{F}_n \in \hat{\mathcal{X}_n}$, such that for each $i \in [n]$, $||F_i - \hat{F}_i||_1 \leq \varepsilon$.

We estimate the Wasserstein-1 distance between $\mathbf{X} \coloneqq (X_1, X_2, \dots X_n)$ and $\hat{\mathbf{X}} \coloneqq (\hat{X}_1, \hat{X}_2, \dots \hat{X}_n)$. Applying Lemma 1 n times with the fact that in one dimension, the Wasserstein distance is equivalent to the $||\cdot||_1$ distance between CDFs we can write:

$$W_1(\mathbf{X}, \hat{\mathbf{X}}) \leq \sum_{i=1}^{n} W_1(X_i, \hat{X}_i) \leq n\varepsilon$$

Using the fact that f is Lipschitz and Lemma 2 we have:

$$W_1(f(\mathbf{X}), f(\hat{\mathbf{X}})) \leq LW_1(\mathbf{X}, \hat{\mathbf{X}})$$

We write $(Y_1, \dots Y_m) = \mathbf{Y} = f(\mathbf{X})$ and $(\hat{Y}_1, \dots \hat{Y}_m) = \hat{\mathbf{Y}} = f(\hat{\mathbf{X}})$. Applying Lemma 3 for each $i \in [m]$ we obtain:

$$W_1(Y_i, \hat{Y}_i) \leq W_1(\mathbf{Y}, \hat{\mathbf{Y}})$$

The whole approximation gives us:

$$W_1(Y_i, \hat{Y}_i) \leq nL\varepsilon$$

Since Y_i and $\hat{Y}_i$ are one-dimensional, the Wasserstein-1 distance is the $||\cdot||_1$ distance between the CDFs. This shows that $\hat{\mathcal{Y}}_i$ forms a $nL\varepsilon$-covering of $\mathcal{Y}_i$. □

5 Practical considerations

This section explains how we implement a solution utilising Algorithm 1.

5.1 Bounding the Lipschitz constant of f

Calculating the exact Lipschitz constant of a neural network is NP-hard [20]. We use the LipSDP software introduced in [8] which estimates the upper bound of the Lipschitz constant for f. The Lipschitz constant returned by LipSDP is in $||\cdot||_2$ norm and we find the bound for the Lipschitz constant in $||\cdot||_1$ using the inequality between power means.

5.2 Inaccuracy of empirical CDFs

We briefly describe how output CDFs are approximated with ECDFs and provide an accuracy bound when the input distributions have bounded supports.

Definition 6 (Empirical cumulative distribution function). *Let X be a random variable and let $x^{(1)}, \ldots, x^{(s)} \sim X$ be s i.i.d. samples. The* empirical CDF *(ECDF) of X based on these samples is:*

$$\widehat{F}_{X,s}(t) \;:=\; \frac{1}{s}\sum_{k=1}^{s} \mathbf{1}\{\, x^{(k)} \le t \,\}, \qquad t \in \mathbb{R},$$

In Algorithm 1, instead of first constructing the $\widehat{F}_{\mathbf{Y},s}$ and then considering its marginals, we construct the empirical marginal distributions $F_{Y_i,s}$ straight away from the samples.

Error bound The strong law of large numbers implies that as the number of samples approaches infinity, the empirical distribution of a variable almost surely approaches the true value of the distribution. Due to Glivenko-Cantelli theorem, the convergence is uniform. The Dworetzky-Kiefer-Wolfowitz inequality allows us to estimate the error between an ECDF and a CDF given the number of samples:

Theorem 2 (Dworetzky-Kiefer-Wolfowitz inequality). *For an empirical distribution function from n samples $F_{X,n}$ and a CDF F_X, it holds for any $\varepsilon > 0$ that:*

$$\mathbb{P}\left(\sup_{x\in\mathbb{R}} |F_{X,n}(x) - F_X(x)| \ge \varepsilon\right) \le Ce^{-2n\varepsilon^2}$$

where C is a constant ([15] proves that the inequality holds for $C = 2$).

Incorporating DKW into the result of Theorem 1 If all input p-boxes have bounded support, the following result holds:

Theorem 3. *Consider a neural network f with n independent inputs, m outputs, an $||\cdot||_1$-norm Lipschitz constant of at most L, and ε-coverings for each input. Given that the marginal output sets have bounded support, for each $i \in [m]$ denoted by $[a_i, b_i]$, then with a probability of at least $1 - 2e^{-2s\varepsilon'^2}$ for any $\varepsilon' > 0$, Algorithm 1 produces $nL\varepsilon + (b_i - a_i)\varepsilon'$ coverings of each marginal output set.*

The case of unbounded support If some of the input p-boxes have unbounded support, the above does not hold. However, DKW can be used to derive additional guarantees in $||\cdot||_\infty$.

6 Constructing coverings of p-boxes

To perform Algorithm 1, finite ε-coverings of each input p-box are required. We present two approaches to constructing coverings:

Step-function coverings. As step functions are dense in $L_1(\mathbb{R})$, it is possible to use them to construct coverings of p-boxes. To generate a covering of $\mathcal{X}$ consisting of staircase functions, we overlay a grid with rectangles of size $x \times y$ onto $\mathcal{X}$, and enumerate all staircase functions which turn only at the points of the grid and are entirely contained within $\mathcal{X}$. This approach may not produce a covering if the bounding distributions of $\mathcal{X}$ have heavy tails - however, in appendix B we show that posing a restriction on the tails is sufficient for this procedure to generate a covering with $\varepsilon \coloneqq x + y$.

Coverings for parametric p-boxes. When a p-box is defined by a parameter range within a known distribution family, in order to construct a covering it usually suffices to consider a set of distributions from this family with varied parameters. For example, consider the parametric p-box:

$$\mathcal{X} = \left\{ N(\mu, \sigma^2) \,\middle|\, \mu \in [\mu_1, \mu_2] \right\}$$

with fixed σ. Choosing $n+1$ distributions with equidistant means $\mu_i = \mu_1 + i\Delta$ with $\Delta = (\mu_2 - \mu_1)/n$ yields a covering $\{N(\mu_i, \sigma^2)\}$ that forms a $(\Delta/2)$-covering. This strategy was used in our experiments, where it provides a tight and scalable approximation of the full p-box.

These constructions offer a trade-off between generality and efficiency: step-function coverings are applicable for arbitrary p-boxes with some restrictions on the tails but are poorly scalable due to combinatorial explosion - we were unable to generate step-function coverings with a precision high enough to be used in our experiments. Parametric coverings are easy to generate and scale well with precision, but they convey less information. In our experiments we only consider parametric p-boxes and parametric coverings due to the scalability advantage.

7 Property verification

In this section we show how to estimate the probability of safety property satisfaction. We bound the probability of an output $\mathbf{y}$ satisfying a safety property P: $A\mathbf{y} < v$ for some matrix A and a vector v.

Guaranteed probabilistic bounds Given a probabilistic output set $\mathcal{Y}$, the guaranteed probabilistic bounds for P are defined as an interval $[lb, ub]$ where:

$$lb = \inf_{\mathbf{Y} \in \mathcal{Y}} (\mathbb{P}(A\mathbf{y} < v | \mathbf{y} \sim \mathbf{Y}))$$
$$ub = \sup_{\mathbf{Y} \in \mathcal{Y}} (\mathbb{P}(A\mathbf{y} < v | \mathbf{y} \sim \mathbf{Y}))$$

Estimating probabilistic bounds By running Algorithm 1, we obtain sets of samples from output vectors $\mathbf{Y}_1, \mathbf{Y}_2, \ldots$. For a single random vector $\mathbf{Y}$, the probability of it satisfying P is estimated by:

$$\mathbb{P}(A\mathbf{y} < v | \mathbf{y} \sim \mathbf{Y}) \sim \frac{\text{number of samples from } \mathbf{Y} \text{ that satisfy } P}{s}$$

Aggregating these estimates for the output vectors $\mathbf{Y}_1, \mathbf{Y}_2, \ldots$ we obtain approximate probability bounds for satisfying P.

We obtain only approximate probability bounds because of the following:

1. *Approximation error of empirical distributions.* Determining the probability of $\mathbf{Y}$ satisfying P is equivalent to estimating the CDF of $A\mathbf{Y}$ at v. Effectively, we calculate the ECDF of $A\mathbf{Y}$ at v, and the approximation error of the ECDF can be estimated with Multivariate DKW inequality:

   Theorem 4. *For a m-dimensional empirical distribution function from s samples $F_{s,\mathbf{X}}$ and a true CDF $F_{\mathbf{X}}$, it holds for any $\varepsilon > 0$ that:*

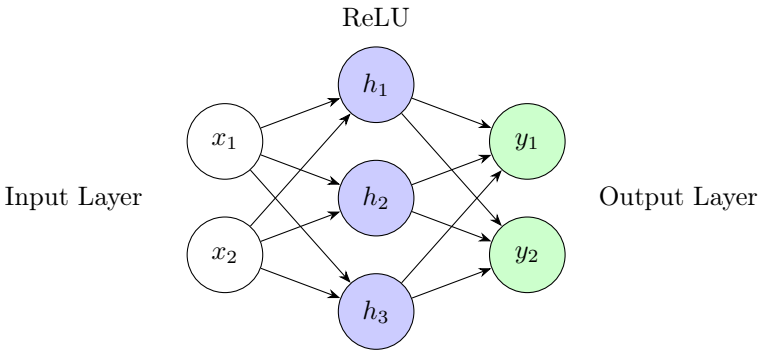
   $$\mathbb{P}(\sup_{\theta \in \mathbb{R}^m} |F_{s,\mathbf{X}}(\theta) - F_{\mathbf{X}}(\theta))| > \varepsilon) \leq m(s+1)e^{-2s\varepsilon^2}$$

2. *Not considering a covering of the whole output space in $||\cdot||_\infty$* If we had a set of vectors $\hat{\mathcal{Y}}$ which formed a covering of $\mathcal{Y}$ in $||\cdot||_\infty$, then for an output vector $\mathbf{Y}$ there would exist $\hat{\mathbf{Y}} \in \hat{\mathcal{Y}}$ such that $||F_{\hat{\mathbf{Y}}} - F_{\mathbf{Y}}||_\infty < \varepsilon$. Specifically, this would allow us to explicitly derive the maximal difference between $\mathbb{P}(A\mathbf{y} < v | \mathbf{y} \sim \mathbf{Y})$ and $\mathbb{P}(A\hat{\mathbf{y}} < v | \mathbf{y} \sim \hat{\mathbf{Y}})$. However, since the vectors output by Algorithm 1 are not guaranteed to form a covering of $\mathcal{Y}$ in $||\cdot||_\infty$, it is possible that there exists an output vector $\mathbf{Y}$, for which the probability of satisfying P lies outside of the property satisfaction bounds that we derived.

Due to the points above, the property satisfaction bounds that we provide in the experiments section are estimates and not guaranteed probabilistic bounds.

8 Experiments

Toy example We consider a neural network f:



with weights given by $W_1 = \begin{bmatrix} 1.0 & -1.0 \\ 1.0 & 1.0 \\ -1.0 & 2.0 \end{bmatrix}$, $b_1 = \begin{bmatrix} 0.0 \\ 0.0 \\ 0.0 \end{bmatrix}$, $W_2 = \begin{bmatrix} 1.0 & -1.0 & 1.0 \\ 1.0 & -1.0 & 2.0 \end{bmatrix}$, $b_2 = \begin{bmatrix} 0.0 \\ 0.0 \end{bmatrix}$. Both inputs are parametric p-boxes of Gaussians with uncertain

mean: $N(\mu \in [0,1], \sigma = 1)$. We generate 0.005-coverings for each input. The Lipschitz constant is bounded from above by 7.07. Running Algorithm 1 with $s = 1000$ samples we obtain 0.07-coverings of the marginal output sets. Figure shows how these coverings compare to the p-boxes obtained with the first method from [10], DSI propagation. However, the software from [10] treats all p-boxes non-parametrically, which can cause some of the discrepancy between the results.

As Gaussian distributions have unbounded supports, Theorem 3 is inapplicable here. However, DKW inequality implies that with a probability of at least 0.9, each empirical distribution is at most 0.001 away from the true distribution in terms of $||\cdot||_\infty$.

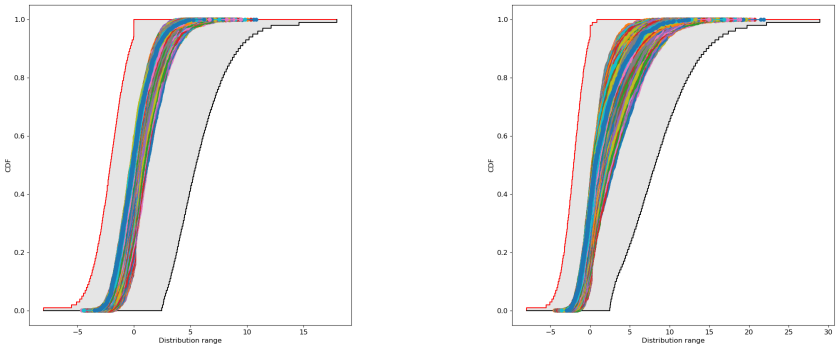


Fig. 1. Output coverings produced by Algorithm 1 (colorful), compared with output p-boxes produced by DSI propagation (grey). The left figure shows the results for the first dimension of the output, the right one those for the second dimension.

8.1 ACAS Xu

We reproduce the property verification for the ACAS Xu example done in [10]. ACAS Xu is a set of 46 networks, each with 5 inputs and 5 outputs. As the complexity of Algorithm 1 is exponential with respect to the input size, $n = 5$ limits the covering precision that we can achieve in this experiment. The ACAS Xu networks' Lipschitz constant (L) estimates obtained by LipSPD are of order 10^4. This prevents us from providing an accurate theoretical guarantee on our results - the accuracy of the output coverings depend on the value of L and the accuracy of the input coverings, both of which are limited. Therefore, there can exist output distributions far away in terms of the $||\cdot||_1$ distance from any output distribution obtained by algorithm 1. However, LipSPD calculates L without taking into account input constraints, and the points which force L to be large can be very few. This lets us hypothesise that choosing a method of

approximating L that incorporates the local input constraints can yield a lower value of L. Thus, we conduct the experiment regardless of the lack of theoretical guarantee on the results.

The exponential complexity of Algorithm 1 forces us to represent input uncertainty with parametric p-boxes, as the covering sizes of non-parametric p-boxes are too large.

Setting Inputs to each ACAS Xu network are parametrised by 2 vectors, ub and lb. Initially, each scalar input is a Gaussian with mean $\mu_i = (ub[i] + lb[i])/2$ and standard deviation $\sigma_i = (ub[i] - \mu_i)/3$. Uncertainty is added by representing each input as a Gaussian parametric p-box defined by $N(\mu \in [\mu_i - 0.001 \cdot r[i], \mu_i + 0.001 \cdot r[i]], \sigma = \sigma_i)$ with $r[i] = ub[i] - lb[i]$. We generate 0.00005-coverings for each input. We remark that despite having the same precision, these coverings are of drastically different sizes - respectively, for each input, 3, 21, 21, 2, 2. This gives us an approximate cost of running Algorithm 1 with $s = 1000$ samples at $3 \times 21 \times 21 \times 2 \times 2 \times 1000 \times C = 5292000C = 5 \times 10^6 C$, where C is the cost of passing a single sample through the network. For speeding-up, Algorithm 1 can be parallelised as subsequent iterations of the main loop are independent of each other.

We verify a safety property considered by [10] - $P_2 : y_1 > y_2 \wedge y_1 > y_3 \wedge y_1 > y_4 \wedge y_1 > y_5$ on a number of networks in ACAS Xu. We also recreate this experiment with input p-boxes of the same shape with the software from [10] and compare the results in Table 1.

A single run of Algorithm 1 with 12 threads on an ACAS Xu network on an Apple M3 Pro chip terminates in under one minute. DSZ propagation terminates in around one minute as well.

Table 1. Probability bounds for the ACAS Xu example.

Prop	Net	DSZ propagation from [10] *Experiment A*	Empirical property verification *Experiment B*
		$\mathbb{P}$	$\mathbb{P}$
2	1-6	[0, 0.03]	[0, 0.001]
2	2-2	[0, 0.09]	[0.022, 0.054]
2	2-9	[0, 0.09]	[0, 0.007]
2	3-1	[0.010, 0.102]	[0.029, 0.066]
2	3-6	[0.013, 0.141]	[0.018, 0.058]
2	3-7	[0, 0.176]	[0, 0.014]
2	4-1	[0, 0.071]	[0, 0.010]
2	4-7	[0.006, 0.136]	[0.028, 0.062]
2	5-3	[0, 0.051]	[0, 0.001]

Results In each case, the probabilistic bounds obtained via B are tighter than in A. In certain cases, the difference is larger than one order of magnitude. However, this is not only due to the conservatism of A, as discussed in the next paragraph.

Differences between the experiments Table 2 shows the differences between the setting of experiments A and B. The first difference can be resolved by setting a high discretization factor in A and propagating more samples in B, however we were neither able to quantify nor to alleviate the rest.

#	A	B
1	Precision depends on the input discretization	Precision depends on the number of samples propagated
2	Considers non-parametric p-boxes	Considers parametric p-boxes (potential loss of information)
3	The probabilistic bounds can be too broad due to conservativeness	The probabilistic bounds can be too tight, as extremal distributions might not have been sampled.

Table 2. Comparison between experiments A and B.

9 Future work

We highlight below the important directions for future work:

1. *Lipschitz constant approximation.* As shown in the ACAS Xu example, the Lipschitz constant estimate provided by LipSDP can be large enough to prevent us from guaranteeing the correctness of the results with Theorem 1. Future work should explore different methods for estimating the Lipschitz constant that incorporate the input constraints.
2. *Relaxations to the theory.* The assumption that the inputs to f are independent rarely holds in reality and lifting it is an important next step.
3. *Parametric vs non-parametric P-boxes.* The approach in [10] could be modified to handle the restriction to parametric p-boxes, as an alternative to designing tractable non-parametric coverings.
4. $||\cdot||_\infty$ *instead of* $||\cdot||_1$. Reproducing Theorem 1 with the infinity norm would allow for integrating DKW into the main result without assumptions on the support. Moreover, in the context of property verification, obtaining a covering of the whole output space in $||\cdot||_\infty$ would allow for providing property satisfaction probabilities with guaranteed error bounds, as explained in section 7.

5. *Coverings using Gaussian mixture distributions.* As shown in [5], Gaussian mixtures can approximate arbitrary probability densities - for any PDF $f \in L_2(\mathbb{R})$ and $\varepsilon > 0$, there exists a Gaussian mixture $\hat{f}$ such that:

$$||\hat{f} - f||_2 \leq \varepsilon$$

A promising direction for future extensions is extending this to obtain a result for approximating arbitrary CDFs with CDFs of Gaussian mixtures, and based on it find a method for constructing coverings with Gaussian mixtures.

6. *Reducing the complexity of Algorithm 1.* For a probability box delimited by $\underline{F}$ and $\overline{F}$, distributions close to $\underline{F}$ and $\overline{F}$ carry more information on the shape of the p-box. In our case, being able to predict which input vectors $\mathbf{X}$ after propagation produce vectors $f(\mathbf{X})$ which are close to the boundaries of the output sets would let us reduce the number of vectors that we propagate. Namely, we would only need to propagate these vectors, as the others would not provide additional information. For a random variable, checking whether after propagation it generates an extremal distribution can be heuristically approximated by its expected value - namely, distributions with lower mean tend to be closer to $\overline{F}$, and conversely. To give an example, in our setting, given that the input vectors are parametric, finding an input vector that is likely to produce an extremal distribution on the first output of f is done by solving:

$$\underset{\mathbf{X}=(X_i \sim F_{\theta_i}),\ (Y_1,\dots Y_m):=\mathbf{Y}\sim f(\mathbf{X})}{argmin} \mathbb{E}(Y_1)$$

for the parameters $\theta_1, \dots \theta_n$. Given a polynomial approximation P of f this can be rewritten as:

$$\underset{\mathbf{X}=(X_i \sim F_{\theta_i}),\ (Y_1,\dots Y_m):=\mathbf{Y}\sim P(\mathbf{X})}{argmin} \mathbb{E}(Y_1)$$

If the distributions that we are considering are Gaussian mixtures and P is of order 1, this is reduced to a linear programming problem. However, for polynomials of higher order and for different distributions, this is a non-trivial problem.

10 Conclusions

To the best of our knowledge, this work is the first study to analyse how to sample from p-box representations and build a verification algorithm around it. Moreover, it provides methods for generating coverings of p-boxes to be used in Algorithm 1. However, due to its exponential complexity, Algorithm 1 has certain limitations - considering neural networks with non-parametric p-boxes as inputs is currently completely intractable, as is running the algorithm on neural networks with larger input dimensions. With the propositions listed in

the future work section, we hope to adapt it to more general input settings, derive guaranteed bounds for property satisfaction and lower the complexity to allow treating larger neural networks.

Acknowledgments. This work was partially supported by the SAIF project, funded by the France 2030 government investment plan managed by the French National Research Agency, under the reference ANR-23-PEIA-0006, and by the 2021 project FARO, funded by Agence de l'Innovation de Défense AID through the Centre Interdisciplinaire d'Etudes pour la Défense et la Sécurité CIEDS.

References

1. Baluta, T., Chua, Z.L., Meel, K.S., Saxena, P.: Scalable quantitative verification for deep neural networks. In: Proceedings of the 43rd International Conference on Software Engineering. p. 312–323. ICSE '21, IEEE Press (2021)
2. Baluta, T., Shen, S., Shinde, S., Meel, K.S., Saxena, P.: Quantitative verification of neural networks and its security applications. In: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security. p. 1249–1264. CCS '19, Association for Computing Machinery, New York, NY, USA (2019)
3. Beer, M., Ferson, S., Kreinovich, V.: Imprecise probabilities in engineering analyses. Mechanical systems and signal processing **37**(1-2), 4–29 (2013)
4. Boopathy, A., Weng, T.W., Chen, P.Y., Liu, S., Daniel, L.: Cnn-cert: an efficient framework for certifying robustness of convolutional neural networks. In: Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence. AAAI'19/IAAI'19/EAAI'19, AAAI Press (2019)
5. Calcaterra, C., Boldt, A.: Approximating with Gaussians (2008), `https://arxiv.org/abs/0805.3795`
6. Enszer, J.A., Lin, Y., Ferson, S., Corliss, G.F., Stadtherr, M.A.: Propagating uncertainties in modeling nonlinear dynamic systems. In: Proceedings of the 3rd International Workshop on Reliable Engineering, Computing, Georgia Institute of Technology, Savannah, GA,: 89. vol. 105 (2008)
7. Fazlyab, M., Morari, M., Pappas, G.J.: Probabilistic verification and reachability analysis of neural networks via semidefinite programming. In: 2019 IEEE 58th Conference on Decision and Control (CDC). pp. 2726–2731. IEEE (2019)
8. Fazlyab, M., Robey, A., Hassani, H., Morari, M., Pappas, G.J.: Efficient and accurate estimation of lipschitz constants for deep neural networks. Curran Associates Inc., Red Hook, NY, USA (2019)
9. Ferson, S., Kreinovich, V., Ginzburg, L., Myers, D., Sentz, K.: Constructing probability boxes and Dempster-Shafer structures. Sandia Report (04 2003)
10. Goubault, E., Putot, S.: A zonotopic Dempster-Shafer approach to the quantitative verification of neural networks. In: Formal Methods: 26th International Symposium, FM 2024, Milan, Italy, September 9–13, 2024, Proceedings, Part I. pp. 324–342. Springer-Verlag, Berlin, Heidelberg (2024)
11. Gray, A., Gopakumar, V., Rousseau, S., Destercke, S.: Guaranteed confidence-band enclosures for pde surrogates. arXiv preprint arXiv:2501.18426 (2025)

12. Henriksen, P., Lomuscio, A.: Efficient neural network verification via adaptive refinement and adversarial search. In: ECAI 2020, pp. 2513–2520. IOS Press (2020)
13. Huang, C., Hu, Z., Huang, X., Pei, K.: Statistical certification of acceptable robustness for neural networks. In: Farkaš, I., Masulli, P., Otte, S., Wermter, S. (eds.) Artificial Neural Networks and Machine Learning – ICANN 2021. pp. 79–90. Springer International Publishing, Cham (2021)
14. Lyu, Z., Ko, C.Y., Kong, Z., Wong, N., Lin, D., Daniel, L.: Fastened crown: Tightened neural network robustness certificates. Proceedings of the AAAI Conference on Artificial Intelligence **34**(04), 5037–5044 (Apr 2020)
15. Massart, P.: The tight constant in the Dvoretzky-Kiefer-Wolfowitz inequality. The Annals of Probability **18**(3), 1269 – 1283 (1990)
16. Monchot, P., Coquelin, L., Petit, S.J., Marmin, S., Le Pennec, E., Fischer, N.: Input uncertainty propagation through trained neural networks. In: Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., Scarlett, J. (eds.) Proceedings of the 40th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 202, pp. 25140–25173. PMLR (23–29 Jul 2023)
17. Păsăreanu, C., Converse, H., Filieri, A., Gopinath, D.: On the probabilistic analysis of neural networks. In: Proceedings of the IEEE/ACM 15th International Symposium on Software Engineering for Adaptive and Self-Managing Systems. pp. 5–8 (2020)
18. Pautov, M., Tursynbek, N., Munkhoeva, M., Muravev, N., Petiushko, A., Oseledets, I.: CC-CERT: A probabilistic approach to certify general robustness of neural networks. Proceedings of the AAAI Conference on Artificial Intelligence **36**, 7975–7983 (06 2022)
19. Pilipovsky, J., Sivaramakrishnan, V., Oishi, M., Tsiotras, P.: Probabilistic verification of relu neural networks via characteristic functions. In: Learning for Dynamics and Control Conference. pp. 966–979. PMLR (2023)
20. Scaman, K., Virmaux, A.: Lipschitz regularity of deep neural networks: analysis and efficient estimation. In: Proceedings of the 32nd International Conference on Neural Information Processing Systems. p. 3839–3848. NIPS'18, Curran Associates Inc., Red Hook, NY, USA (2018)
21. Singh, G., Gehr, T., Mirman, M., Püschel, M., Vechev, M.: Fast and effective robustness certification. In: Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 31. Curran Associates, Inc. (2018)
22. Singh, G., Gehr, T., Püschel, M., Vechev, M.: An abstract domain for certifying neural networks. Proc. ACM Program. Lang. **3**(POPL) (Jan 2019)
23. Walley, P.: Statistical Reasoning with Imprecise Probabilities. Chapman & Hall/CRC Monographs on Statistics & Applied Probability, Taylor & Francis (1991)
24. Webb, S., Rainforth, T., Teh, Y.W., Kumar, M.P.: A statistical approach to assessing neural network robustness. In: 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019. OpenReview.net (2019)
25. Weng, L., Chen, P.Y., Nguyen, L., Squillante, M., Boopathy, A., Oseledets, I., Daniel, L.: PROVEN: Verifying robustness of neural networks with a probabilistic approach. In: Chaudhuri, K., Salakhutdinov, R. (eds.) Proceedings of the 36th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 97, pp. 6727–6736. PMLR (09–15 Jun 2019)

26. Wright, O., Nakahira, Y., Moura, J.M.: An analytic solution to covariance propagation in neural networks. In: International Conference on Artificial Intelligence and Statistics. pp. 4087–4095. PMLR (2024)
27. Zhang, D., Ye, M., Gong, C., Zhu, Z., Liu, Q.: Black-box certification with randomized smoothing: a functional optimization based framework. In: Proceedings of the 34th International Conference on Neural Information Processing Systems. NIPS'20, Curran Associates Inc., Red Hook, NY, USA (2020)
28. Zhang, H., Weng, T.W., Chen, P.Y., Hsieh, C.J., Daniel, L.: Efficient neural network robustness certification with general activation functions. In: Proceedings of the 32nd International Conference on Neural Information Processing Systems. p. 4944–4953. NIPS'18, Curran Associates Inc., Red Hook, NY, USA (2018)
29. Zhang, T., Ruan, W., Fieldsend, J.E.: Proa: A probabilistic robustness assessment against functional perturbations. In: Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2022, Grenoble, France, September 19–23, 2022, Proceedings, Part III. p. 154–170. Springer-Verlag, Berlin, Heidelberg (2023)

A Proofs of theorems

Proof (of lemma 1). We write:

$$W_1(\mathbf{X},\mathbf{Y}) = \inf_{\gamma\in\Pi} \mathbb{E}\; d\left(\begin{pmatrix}x_1\\x_2\\\vdots\\x_n\end{pmatrix},\begin{pmatrix}y_1\\y_2\\\vdots\\y_n\end{pmatrix}\right)$$

Consider the optimal couplings between X_1 and Y_1 and between $(X_2,\dots X_n)$ and $(Y_2,\dots Y_n)$: γ_1 and γ_2. Then $\gamma = \gamma_1\gamma_2$ is a coupling between all the variables and we have:

$$\begin{aligned}\mathbb{E}_\gamma\; d\left(\begin{pmatrix}x_1\\x_2\\\vdots\\x_n\end{pmatrix},\begin{pmatrix}y_1\\y_2\\\vdots\\y_n\end{pmatrix}\right)&\\ = \mathbb{E}_\gamma\left(d(x_1,y_1) + d\left(\begin{pmatrix}x_2\\\vdots\\x_n\end{pmatrix},\begin{pmatrix}y_2\\\vdots\\y_n\end{pmatrix}\right)\right)&\\ = \mathbb{E}_{\gamma_1} d(x_1,y_1) + \mathbb{E}_{\gamma_2} d\left(\begin{pmatrix}x_2\\\vdots\\x_n\end{pmatrix},\begin{pmatrix}y_2\\\vdots\\y_n\end{pmatrix}\right)&\end{aligned}$$

As γ is some coupling, not necessarily optimal, we have:

$$W_1(\mathbf{X},\mathbf{Y}) \le W_1(X_1,Y_1) + W_1\left(\begin{pmatrix}X_2\\\vdots\\X_n\end{pmatrix},\begin{pmatrix}X_2\\\vdots\\X_n\end{pmatrix}\right)$$

□

Proof (of lemma 2). Observe that for a given coupling γ we have:

$$\mathbb{E}_\gamma \; d\left(\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}, \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}\right)$$

$$= \mathbb{E}_\gamma \left(d(x_1, y_1) + d\left(\begin{pmatrix} x_2 \\ \vdots \\ x_n \end{pmatrix}, \begin{pmatrix} y_2 \\ \vdots \\ y_n \end{pmatrix}\right)\right)$$

$$= \mathbb{E}_\gamma d(x_1, y_1) + \mathbb{E}_\gamma d\left(\begin{pmatrix} x_2 \\ \vdots \\ x_n \end{pmatrix}, \begin{pmatrix} y_2 \\ \vdots \\ y_n \end{pmatrix}\right)$$

Therefore:

$$\mathbb{E}_\gamma \; d\left(\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}, \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}\right) \geq \mathbb{E}_\gamma d(x_1, y_1)$$

Taking the infinum over all couplings between (X_1, Y_1) and $(X_2, \dots X_n), (Y_2, \dots Y_n)$ we get the desired result. It is important to remark that all couplings γ cover all the possible couplings between X_1 and Y_1. This is not difficult, as for any such coupling γ_1, a coupling γ_2 between $(X_2, \dots X_n)$ and $(Y_2 \dots Y_n)$ (independent of γ_1) can be considered and then $\gamma = \gamma_1\gamma_2$ is a valid coupling between (X_1, Y_1) and $(X_2, \dots X_n), (Y_2, \dots Y_n)$. □

Proof (of lemma 3). Fwor a coupling γ between $\mathbf{X}$ and $\mathbf{Y}$ and $\mathbf{x}, \mathbf{y} \sim \gamma$ we have:

$$||f(\mathbf{x}) - f(\mathbf{y})|| \leq L||\mathbf{x} - \mathbf{y}||$$

Therefore:

$$\mathbb{E}_{\mathbf{x},\mathbf{y}\sim\gamma}||f(\mathbf{x}) - f(\mathbf{y})|| \leq L\mathbb{E}_{\mathbf{x},\mathbf{y}\sim\gamma}||\mathbf{x} - \mathbf{y}||$$

Since this holds for any coupling, we can take the infinum and obtain:

$$W_1(f(\mathbf{X}), f(\mathbf{Y})) \leq LW_1(\mathbf{X}, \mathbf{Y})$$

□

Proof (of theorem 3). Theorem 1 yields a covering result on true CDFs - i.e. for a single output with the constrained probabilistic output set on this output having a compact support $[a, b]$, any output CDF F is at most $nL\varepsilon$ away in terms of $||\cdot||_1$ from an output CDF $\hat{F}$ belonging to the propagated input covering.

When propagating empirically, instead of $\hat{F}$ we have an empirical distribution $\hat{F}_s$ estimating $\hat{F}$. By Hölder's inequality, on the interval $[a, b]$ it holds that:

$$||\hat{F}_s - \hat{F}||_1 \leq (b-a)||\hat{F}_n - \hat{F}||_\infty$$

By DKW, for any $\varepsilon' \geq 0$ it holds that:

$$\mathbb{P}\left(||\hat{F}_s - \hat{F}||_1 \geq (b-a)\varepsilon'\right) \leq 2e^{-2s\varepsilon'^2}$$

Therefore, with a probability of at least $1 - 2e^{-2s\varepsilon'^2}$, it holds that:

$$||\hat{F}_s - F||_1 \leq \varepsilon + (b-a)\varepsilon'$$

Deriving this on each output dimension gives us the desired result. □

B Precision of covering a p-box with staircase functions on a grid

We present the method of generating a covering of a p-box with staircase functions on a grid and quantify its precision. Consider a p-box $\mathcal{X}$ bounded by $\underline{F}$ and $\overline{F}$, such that for some $x_l \in \mathbb{R}$ $\int_{-\infty}^{x_l} \overline{F}(t)dt$ is close to 0 and for some $x_r \in \mathbb{R}$, $\int_{x_r}^{\infty}(1 - \underline{F}(t))dt$ is close to 0. Let G be a grid of points in $\mathbb{R}^2$, with rectangles of size $x \times y$ for some $x, y \in \mathbb{R}$ - that is, $G = \{(a, b)|\exists n, m \in \mathbb{Z} : a = mx, b = ny, x_l \leq a \leq x_r, 0 \leq b \leq 1\}$. We show that the set $\hat{\mathcal{X}}$ of staircase functions that turn only on points belonging to G and are entirely contained within $\mathcal{X}$ is a covering of $\mathcal{X}$.

The assumption on the tails of $\underline{F}$ and $\overline{F}$ allows us to disregard everything outside of $[x_l, x_r]$ - all the functions in $\hat{\mathcal{X}}$ are identically zero before x_l and identically one after x_r. Therefore, given $\mathbf{X} \in \mathcal{X}$ and $\hat{\mathbf{X}} \in \hat{\mathcal{X}}$, the distance between $F_{\mathbf{X}}$ and $F_{\hat{\mathbf{X}}}$ on $(-\infty, x_l) \cup (x_r, \infty)$ is at most $\int_{-\infty}^{x_l} \overline{F}(t)dt + \int_{x_r}^{\infty}(1 - \underline{F}(t))dt$.

We prove an auxilliary statement:

Theorem 5. *The set of all staircase functions that turn only on points belonging to G is a ε-covering of the set of all CDFs defined on $[x_l, x_r]$, and $\varepsilon = O(x + y)$.*

Proof. Consider an arbitrary CDF F defined on $[x_l, x_r]$. For simplicity, we denote $a_0 \coloneqq x_l, a_1 \coloneqq x_l + x, \ldots a_m \coloneqq x_l + mx$. We approximate it with a staircase function F_1 defined as:

$$F_1(t) = \begin{cases} F(a_i), & \text{if } t \in [a_i, a_{i+1}) \text{ for each } i \in 0, \ldots m-1 \\ F(a_m) & \text{if } t = a_m \end{cases}$$

Since F is an increasing functions, we can bound the $||\cdot||_1$ difference between F and F_1 on each interval $[a_i, a_{i+1}]$ - namely, for each $i \in 0, \ldots m-1$:

$$\int_{a_i}^{a_{i+1}} |F_1(t) - F(t)|dt \leq \int_{a_i}^{a_{i+1}} |F(a_i) - F(a_{i+1})|dt = x(F(a_{i+1}) - F(a_i))$$

Summing those differences up, we get:

$$\int_{a_0}^{a_m} |F_1(t) - F(t)|dt \leq x \sum_{i=0}^{m-1} F(a_{i+1}) - F(a_i) \leq x$$

Now, we approximate F_1 with a staircase function F_2 going only on the grid. Define a function round(t), that rounds a value in $[0, 1]$ to the closest mutliple of y, rounding down if it is halfway between 2 values. Then, F_2 is defined as:

$$F_2(t) = \text{round}(F_1(t))$$

By definition of the round function, we have that for any $t \in [x_l, x_r]$, $|F_2(t) - F_1(t)| \leq \frac{y}{2}$. Therefore:

$$||F_2 - F_1||_1 \leq \frac{y(x_r - x_l)}{2}$$

And:

$$||F_2 - F||_1 \leq ||F_2 - F_1||_1 + ||F_1 - F||_1 \leq x + \frac{y(x_r - x_l)}{2} = O(x + y)$$

□

We generalise this result to hold with the additional restriction of staying within $\mathcal{X}$:

Theorem 6. *Given the setting and notations of theorem 5, consider $F \in \mathcal{X}$. Then, if $\hat{\mathcal{X}}$ is non-empty, there exists a grid staircase $F_3 \in \mathcal{X}$ such that $||F_3 - F_2||_1 \leq (x_r - x_l)y$ and*

Proof. Assume that $\hat{\mathcal{X}}$ is non-empty. We consider all the intervals $I_i \coloneqq [a_i, a_{i+1})$ for $i \in 0, \dots m-1$. On each interval I_i one of three things can happen:

1. F_2 is entirely within $\mathcal{X}$,
2. certain values of F_2 are above corresponding values of $\overline{F}$, which implies that $F_2(a_i) > \overline{F}(a_i)$,
3. certain values of F_2 are below corresponding values of $\underline{F}$, which implies that $lim_{a \to a_{i+1}} F_2(a) < \underline{F}(a_{i+1})$.

Note that if situations 2 and 3 occur at the same time, it means that $\hat{\mathcal{X}}$ is empty, as all grid points with the y-coordinate equal to $F_2(a_i)$ are outside of $\mathcal{X}$.

We consider a staircase function F_3 constructed the following way - on each interval I_i:

1. If situation 1 occurs, F_3 is identically equal to F_2 on this interval,
2. if situation 2 occurs F_3 is F_2 shifted down by y on this interval,
3. if situation 3 occurs, F_3 is F_2 shifted up by y on this interval.

It remains to show that F_3 is still increasing, and contained within $\mathcal{X}$. There are multiple ways of this happening: for example, at some interval I_i, F_3 is F_2 shifted up, and on the next interval F_3 is F_2 shifted down. However, all of those situations reduce to $\hat{\mathcal{X}}$ being empty. To show that F_3 is within $\mathcal{X}$ we remark that shifting F_2 up or down on an interval I_i, we switch from approximating F from above or from below. If we shift, that means that one of these approximations is outside of the $\mathcal{X}$ on I_i. Therefore, if the shift results in the other approximation also being outside of $\mathcal{X}$ on I_i, then $\mathcal{X}$ admits no grid staircases inside it, so $\hat{\mathcal{X}}$ is empty.

By construction of F_3, on each interval I_i the difference between F_3 and F_2 in terms of $||\cdot||_1$ is at most xy. Summing this up, we obtain the desired result:

$$||F_3 - F_2||_1 \leq (x_r - x_l)y$$

□

Combining the results of Theorems 5 and 6 we obtain that for any $F \in \mathcal{X}$ there exists a grid staircase function $F_3 \in \mathcal{X}$ such that $||F - F_3||_1 = O(x + y)$.