Computing
DOI 10.1007/s00607-011-0182-8

A generalization of p-boxes to affine arithmetic

Olivier Bouissou · Eric Goubault · Jean Goubault-Larrecq · Sylvie Putot

Received: 18 March 2011 / Accepted: 2 December 2011

Abstract We often need to deal with information that contains both interval and probabilistic uncertainties. P-boxes and Dempster–Shafer structures are models that unify both kind of information, but they suffer from the main defect of intervals, the wrapping effect. We present here a new arithmetic that mixes, in a guaranteed manner, interval uncertainty with probabilities, while using some information about variable dependencies, hence limiting the loss from not accounting for correlations. This increases the precision of the result and decreases the computation time compared to standard p-box arithmetic.

Keywords Affine arithmetic · P-boxes · Dempster–Shafer structures

Mathematics Subject Classification (2010) 60A86 · 65G30 · 65G50 · 65C50

The authors have presented the results of this paper during the SCAN 2010 conference in Lyon, September 2010. This work is partially funded by the ANR project ANR09BLAN034502.

O. Bouissou (✉) · E. Goubault · S. Putot
CEA Saclay Nano-INNOV, Point Courrier 174, 91191 Gif-sur-Yvette Cedex, France
e-mail: olivier.bouissou@cea.fr

E. Goubault
e-mail: eric.goubault@cea.fr

S. Putot
e-mail: sylvie.putot@cea.fr

J. Goubault-Larrecq
LSV, ENS Cachan, 61 avenue du président Wilson, 94230 Cachan, France
e-mail: goubault@lsv.ens-cachan.fr

Published online: 04 January 2012

1 Introduction

In program analysis, a difficulty is to properly handle the inputs of the program. They are often not exactly known, but should rather be considered as being in a set of possible values: this introduces interval uncertainty as all values within this set are possible inputs. However, it is often the case that we have more information on the inputs than the mere fact that they belong to some set. In particular, we often know, in addition to the bounds on the inputs, partial information about the probabilities of different values within this set. P-boxes [5,6] and Dempster–Shafer structures [16] are widely used to propagate both probabilistic information and interval uncertainty. However, arithmetic rules on these structures are mostly used between structures representing either independent random variables, or variables with unknown dependency.

This makes the usual p-box arithmetic not well adapted when many numerical computations occur between quantities (values of the program variables for instance) that cannot be considered independent, but whose dependency cannot be determined easily. We emphasize this statement below.

Running example We consider the classical, order 2 Butterworth filter:

$$y_{n+2} = \frac{(2(c^2-1)y_{n+1} - (c^2 - \sqrt{2}c + 1)y_n + c^2 x_{n+2} - 2c^2 x_{n+1} + c^2 x_n)}{c^2 + \sqrt{2}c + 1}.$$

We set $y_0 = y_1 = 0$, with inputs $x_n \in [-1, 1]$ and $c = 10$. The output y_n is bounded for any number n of iterations.

Suppose now that we know that each input is given as an independent random variable, uniform on $[-1, 1]$. The nth output of this linear recursive filter has the probability law of a sum of n independent, uniform random variables, hence converging by the central limit theorem to a Gaussian distribution. A Matlab simulation allows bounding the output for 15 iterations by $[-3.25, 3.25]$, and shows that it is with high probability between -2 and 2. We can wonder how to determine guaranteed knowledge on the distribution of y_n, that a simulation cannot give. We tried this example using p-boxes or Dempster–Shafer structures (DS in short), with for each input a discretization of a uniform law between -1 and 1. We used independent arithmetic between the variables x_n, x_{n-1} and x_{n-2}, and arithmetic with unknown dependency for y_{n-1} and y_{n-2}. After 15 filter iterations, we could only bound y_n between -233.10^3 and 233.10^3, and the over-approximation of the density function we obtain is very coarse: the correlations between variables are not used.

In this article, we define a new arithmetic that combines *affine arithmetic* to propagate the linear relations between random variables and *p-box arithmetic* to over-approximate the probability distribution resulting from a given operation. To sum up, we express a random variable as a linear combination of independent DS (the inputs) and not-necessarily-independent DS (created by non-linear operations), and we use DS arithmetic to combine them. We show that this arithmetic largely increases precision and reduces computation time for the propagation of p-boxes through linear and non-linear computations.

Contents Section 2 recalls basic notions about affine arithmetic and p-boxes. In Sect. 3, we present our new arithmetic on probabilistic affine forms, in Sect. 4, we explain how it can be used to the analysis of rounding errors. Section 5 presents experimental results, and we conclude by some perspectives.

2 Basics

The set of all closed intervals over $\mathbb{R}$ is denoted $\mathbb{I} = \{[a, b] \mid a \leq b\}$. Intervals are denoted by bold symbols, e.g. $\boldsymbol{x}$. For $\boldsymbol{x} \in \mathbb{I}$, we denote $\overline{\boldsymbol{x}}$ its upper bound and $\underline{\boldsymbol{x}}$ its lower bound.

2.1 Perturbed affine forms

Affine arithmetic [3] is an extension of interval arithmetic in which sets of values are parametrized by their dependence on some inputs. Each quantity is represented as an affine combination of so-called *noise symbols* ε_i whose values are uncertain between -1 and 1: $\hat{x} \stackrel{def}{=} \alpha_0^x + \sum_{i=1}^n \alpha_i^x \varepsilon_i$, with $\alpha_i^x \in \mathbb{R}$ for all i. The same noise symbol can be shared by several forms, indicating correlations.

The set of values that some quantity x defined by an affine form $\hat{x}$ can take is in the interval $[\alpha_0^x - \sum_{i=1}^n |\alpha_i^x|, \alpha_0^x + \sum_{i=1}^n |\alpha_i^x|]$, and the set of values that the quantities $x, y, z, \ldots$ can take is a zonotope, that is a center-symmetric polytope, whose faces are all center-symmetric.

In the context of abstract-interpretation based static analysis and validation of programs, we introduced perturbed affine forms [8,9]. In these perturbed affine forms, we made the difference between two kinds of noise symbols: the ε_i, directly related to an (uncertain) input value or parameter, and those, the η_j, that express an uncertainty in the analysis (loss of relation due to nonlinear computations for instance), writing $\hat{x} = \alpha_0^x + \sum_{i=1}^n \alpha_i^x \varepsilon_i + \sum_{j=1}^m \beta_j^x \eta_j$. This allowed us to define a functional abstraction, i.e. we directly represent a set of functions $f : \mathbb{R}^n \to \mathbb{R}^p$ as an (affine) input-output relation. In Sect. 3, in a similar way we will also use two kind of symbols, which will be random variables: the ε_i that are considered independent from one another, and the η_j that have unknown dependency to the others.

Affine forms are closed under affine transformations: for $\lambda \in \mathbb{R}$,

$$\hat{x} + \lambda\hat{y} = \alpha_0^x + \lambda\alpha_0^y + \sum_{i=1}^{n}(\alpha_i^x + \lambda\alpha_i^y)\varepsilon_i + \sum_{j=1}^{m}(\beta_j^x + \lambda\beta_j^y)\eta_j.$$

Multiplication creates a new symbol η_{m+1} associated to the non-linear part:

$$\hat{x} \times \hat{y} = \alpha_0^x\alpha_0^y + \frac{R}{2} + \sum_{i=1}^{n}(\alpha_0^x\alpha_i^y + \alpha_i^x\alpha_0^y)\varepsilon_i + \sum_{j=1}^{m}(\alpha_0^x\beta_j^y + \beta_j^x\alpha_0^y)\eta_j + T\eta_{m+1}$$

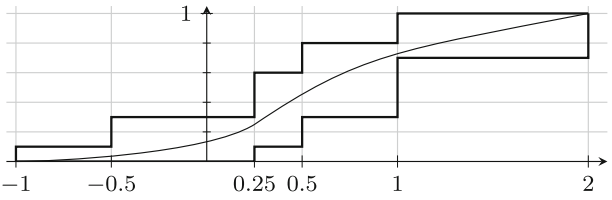


Fig. 1 A p-box (*thick lines*) and a CDF (*thin line*) included in this p-box

with $R = \sum_{i=1}^{n} |\alpha_i^x \alpha_i^y| + \sum_{j=1}^{m} |\beta_j^x \beta_j^y|$ and

$$T = \sum_{i=1}^{n}\sum_{j=1}^{m} | \alpha_i^x \beta_j^y + \beta_j^x \alpha_i^y | + \sum_{i=1}^{n}\sum_{j>i}^{n} | \alpha_i^x \alpha_j^y + \alpha_j^x \alpha_i^y | + \sum_{i=1}^{m}\sum_{j>1}^{m} | \beta_i^x \beta_j^y + \beta_j^x \beta_i^y | + \frac{1}{2} R \ .$$

2.2 Arithmetic operations on Dempster–Shafer structures and p-boxes

An interval based Dempster–Shafer structure [16] (DS in short) is a finite set of closed intervals (named focal elements) associated with a probability. DS structures thus represent real variables whose values can be obtained by first probabilistically picking an interval (given an associated random variable over the real number intervals), and then picking a value within this interval. In this article, we write a DS structure d as $d = \{\langle \boldsymbol{x_1}, w_1\rangle, \langle \boldsymbol{x_2}, w_2\rangle, \ldots, \langle \boldsymbol{x_n}, w_n\rangle\}$, where $\boldsymbol{x_i} \in \mathbb{I}$ is a closed non-empty interval and $w_i \in]0, 1]$ is the associated probability, with $\sum_{k=1}^{n} w_k = 1$.

A set of cumulative distribution functions can be represented by two non-decreasing functions $\underline{P}$ and $\overline{P}$ such that $\underline{P}$ is left-continuous, $\overline{P}$ is right-continuous and $\forall x,\ \underline{P}(x) \leq \overline{P}(x)$. Such a pair $[\underline{P}, \overline{P}]$ is called a p-box [5] and encloses all CDFs P that satisfy $\forall x,\ \underline{P}(x) \leq P(x) \leq \overline{P}(x)$. We will only consider here *discrete* p-boxes, where $\underline{P}$ and $\overline{P}$ are step-functions.

We refer the reader to [5] for the correspondence between discrete p-boxes and DS structures; we denote ζ the function that converts a DS to a p-box, and δ its right-inverse, that converts a discrete p-box to a DS, i.e. for all p-box $P = [\underline{P}, \overline{P}]$, we have $P = \zeta \circ \delta(P)$.

Example 1 Let $d_1 = \{\langle [-1, 0.25], 0.1\rangle; \langle [-0.5, 0.5], 0.2\rangle; \langle [0.25, 1], 0.3\rangle; \langle [0.5, 1], 0.1\rangle; \langle [0.5, 2], 0.1\rangle; \langle [1, 2], 0.2\rangle\}$. Then $[\underline{P_2}, \overline{P_2}] = \zeta(d_1)$ is plotted in Fig. 1.

Many techniques have been proposed for p-boxes or DS arithmetic [1,4,7,20]. Most of them produce the same result [14]. Here, we briefly explain the method of Berleant et al. [1] which we used in our implementation. Let two random variables X and Y represented by DS structures $d_X = \{\langle \boldsymbol{x_i}, w_i\rangle,\ i \in [1, n]\}$ and $d_Y = \{\langle \boldsymbol{y_j}, w'_j\rangle,\ j \in [1, m]\}$, and Z be the random variable such that $Z = X + Y$. We want to compute a DS that contains Z (the algorithms for other arithmetic operations are similar). We distinguish

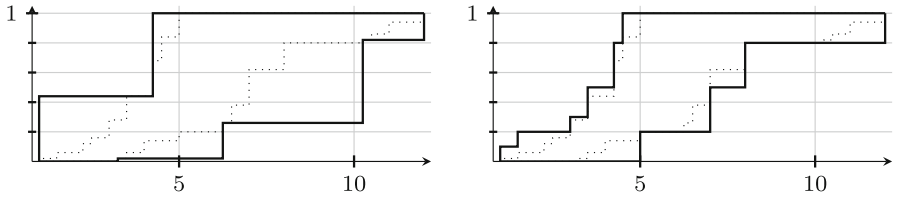


Fig. 2 Results of the sum of d_1 and d_2 from Example 2 in case of independence (*left*) and unknown dependency (*right*)

the case when X and Y are independent (in which case we denote the operation $\oplus$) and when they are not (in which case we denote the operation $+$).
Independent variables If X and Y are independent random variables, then the DS for $Z = X \oplus Y$ is $d_Z = \{\langle \boldsymbol{z_{i,j}}, r_{i,j}\rangle,\ i \in [1,n],\ j \in [1,m]\}$ such that:

$$\forall i \in [1,n],\ j \in [1,m],\ \boldsymbol{z_{i,j}} = \boldsymbol{x_i} + \boldsymbol{y_j} \text{ and } r_{i,j} = w_i \times w'_j\ . \tag{1}$$

The number of focal elements within the DS structures clearly grows exponentially with the number of such operations. In order to keep the computation tractable, we bound the number of focal elements: of course, this introduces some overapproximation.

Example 2 Let X be described by the DS structure d_1 of Example 1 and Y by $d_2 = \{\langle[2,3], 0.2\rangle; \langle[4,6], 0.6\rangle; \langle[4,10], 0.2\rangle\}$. Then, $Z = X \oplus Y$ is described by a DS structure with 18 focal elements. The associated p-box and a reduction of it with at most 4 focal elements are presented on Fig. 2 left, in dotted and thick lines, respectively.

Variables with unknown dependency If the random variables X and Y are not independent, the lower bound $\underline{F_Z}$ of the p-box $[\underline{F_Z}, \overline{F_Z}]$ for Z is obtained by solving the linear programming problem defined by Eq. (2).

$$\begin{aligned} \underline{F_Z}(z) = \textbf{maximize} \quad & \sum\nolimits_{b_{i,j} \le z} r_{i,j} \\ \textbf{such that} \quad & \forall i \in [1,n],\ \sum\nolimits_{j=1}^{m} r_{i,j} = w_i \\ \forall j \in [1,m],\ & \sum\nolimits_{i=1}^{n} r_{i,j} = w'_j \end{aligned} \tag{2}$$

The formula for $\overline{F_Z}$ is similar. We then define $d_Z = \delta([\underline{F_Z}, \overline{F_Z}])$, where δ is the conversion function from p-boxes to DS.

Example 3 We consider the DS d_1 and d_2 (describing X and Y) of Example 2. The p-box for $Z = X + Y$ is given Fig. 2 right (in dotted lines, the p-box obtained for independent variables).

3 Mixing it up: affine arithmetic with probabilistic noise symbols

In this section, we consider a set of uncertain quantities that we want to propagate through a system. We express the linearized dependencies between these quantities using affine arithmetic, while we express the uncertainty in each variable by associating a DS to each noise symbol of the affine forms. In this way, we add structure to DS arithmetic. As in Sect. 2.1, we will define two kinds of noise symbols: the ε_i that are considered independent, and the η_k that cannot be considered as independent from other noise symbols.

Definition 1 (*Probabilistic affine form*) We define a probabilistic affine form for variable x, on n independent noise symbols $(\varepsilon_1, \ldots, \varepsilon_n)$ and m noise symbols $(\eta_1, \ldots, \eta_m)$ with unknown dependency to the others, by a form

$$\hat{x} = \alpha_0^x + \sum_{i=1}^{n} \alpha_i^x \varepsilon_i + \sum_{j=1}^{m} \beta_j^x \eta_j$$

together with n DS $(d_{\varepsilon_1}, \ldots, d_{\varepsilon_n})$ and m DS $(d_{\eta_1}, \ldots, d_{\eta_m})$ describing the possible random variables (of support $[-1, 1]$) for the noise symbols.

A probabilistic affine form $\hat{x}$ represents the set of all random variables X contained in the DS $\gamma(\hat{x})$, called concretization of $\hat{x}$, defined in Definition 2.

Definition 2 (*Concretisation of affine forms as a DS*) Let $\hat{x}$ be a probabilistic affine form, its concretization as a DS is:

$$\gamma(\hat{x}) = \alpha_0^x + \bigoplus_{j=1}^{n} \alpha_j^x d_{\varepsilon_j} + \sum_{k=1}^{m} \beta_k^x d_{\eta_k},$$

where $\sum$ is the sum of DS with unknown dependency, $\bigoplus$ is the sum of independent DS, and we denote αd (resp. $\alpha + d$) where $\alpha \in \mathbb{R}$ and d is a DS $\{\langle \boldsymbol{x_i}, p_i \rangle \mid i = 1, \ldots, q\}$, the result of the multiplication (resp. addition) of a constant by a DS: $\{\langle \alpha \boldsymbol{x_i}, p_i \rangle \mid i = 1, \ldots, q\}$ (resp. $\{\langle \alpha + \boldsymbol{x_i}, p_i \rangle \mid i = 1, \ldots, q\}$).

The interest of affine forms is to be able to represent affine relations that hold between uncertain quantities. We still have this representation, except only *imprecise* affine relations hold, as can be shown in the example below.

Example 4 Let $\hat{x_1} = 1 + \varepsilon_1 - \eta_1$, $\hat{x_2} = -\frac{1}{2}\varepsilon_1 + \frac{1}{4}\eta_1$, $d_{\varepsilon_1} = \{\langle [-1, 0], \frac{1}{2} \rangle, \langle [0, 1], \frac{1}{2} \rangle\}$, $d_{\eta_1} = \{\langle [-\frac{1}{10}, 0], \frac{1}{2} \rangle, \langle [0, \frac{1}{10}], \frac{1}{2} \rangle\}$, Then $\hat{x_1} + 2\hat{x_2} = 1 - \frac{1}{2}\eta_1$, with $d = d_{x_1+2x_2} = \{\langle [\frac{19}{20}, 1], \frac{1}{2} \rangle, \langle [1, \frac{21}{20}], \frac{1}{2} \rangle\}$. Thus the lower probability that $x_1 + 2x_2 \leq \frac{21}{20}$ is 1; and the upper probability that $x_1 + 2x_2 < \frac{19}{20}$ is 0. But for instance, $x_2 + 2x_2 \leq 1$ has upper probability $\frac{1}{2}$ and lower probability 0 and is thus an imprecise relation.

For affine arithmetic operations, there is nothing new - the operation is exactly the same on the perturbed affine form part as in Sect. 2.1, and DS structures attached to symbols are not modified.

For non-linear operations, we can rely on the affine form calculus, but instead of only bounding the non-linear part of the multiplication of the affine forms, we use the available calculus on DS to form a correct DS representing this non-linear part. In this computation, some noise symbols are independent to others, and we use carefully, alternatively, independent or unknown dependency versions of the operations on DS.

We will need in the sequel to compute the square of a DS $x = \{\langle \mathbf{x}_i, w_i \rangle \mid i = 1, \ldots, p\}$: it can be checked that it is the DS $x^2 = \{\langle \mathbf{x}_i^2, w_i \rangle \mid i = 1, \ldots, p\}$.

Definition 3 Let $\hat{x}$ and $\hat{y}$ be two probabilistic affine forms on $n\varepsilon_i$ and $m\eta_j$ noise symbols. The probabilistic affine form $\hat{z} = \hat{x} \times \hat{y}$ is composed of an affine form defined as in Sect. 2.1. The DS structures for the existing noise symbols are unchanged; the new symbol η_{m+1} is considered with unknown dependency to the other symbols (characterization of this dependency is left for future work), and has an associated DS $d_{\eta_{m+1}}$ defined by:

$$
\begin{aligned}
d_{\eta_{m+1}} = \tfrac{1}{T} \Bigg(& \sum_{i=1}^{n} \sum_{j=1}^{m} (\alpha_i^x \beta_j^y + \beta_j^x \alpha_i^y) d_{\epsilon_i} \times d_{\eta_j} \\
& + \sum_{i=1}^{n} \sum_{j>i}^{n} (\alpha_i^x \alpha_j^y + \alpha_j^x \alpha_i^y) d_{\epsilon_i} \otimes d_{\epsilon_j} \\
& + \sum_{i=1}^{m} \sum_{j>i}^{m} (\beta_i^x \beta_j^y + \beta_j^x \beta_i^y) d_{\eta_i} \times d_{\eta_j} \\
& + \frac{1}{2} \left(\bigoplus_{i=1}^{n} \alpha_i^x \alpha_i^y {d_{\epsilon_i}}^2 + \sum_{j=1}^{m} \beta_j^x \beta_j^y {d_{\eta_j}}^2 \right) \Bigg)
\end{aligned}
$$

with T and R as defined in Sect. 2.1.

The correctness of the operations on probabilistic affine forms stems directly from the correctness of affine forms arithmetic and DS arithmetic.

Let us remark that, although affine arithmetic usually increases the computation time compared to interval arithmetic, in our case we decrease the complexity of arithmetic operations by using affine forms. Actually, we delay most of the DS arithmetic to the concretization function, and we can use the arithmetic of independent DS structures instead of arithmetic with unknown dependency, which is very costly.

4 Modeling rounding errors

Suppose we have a real uncertain quantity x given by a probabilistic affine form, i.e. an affine form $\hat{x}$ together with a DS structure for each noise symbol. We also want to model the error due to rounding to floating-point numbers, still with affine arithmetic, as in [10]. A quantity x is now modeled by a triplet (f^x, r^x, e^x) of its floating-point value, its real value and its rounding error. We defined in [10] transfer functions for the propagation error through computations with this model. It is out of scope here to

detail them in the context of p-boxes, and most of these functions will be transposed directly, so we only discuss the creation of a new rounding error term.

We denote $f(r^x)$ the nearest floating-point value to real value r^x, and $e(r^x)$ the associated rounding error $e(r^x) = f(r^x) - r^x$. With normalized floating-point numbers and rounding mode to the nearest, we can bound the error committed when rounding the real value r^x to $f(r^x)$ by

$$|f(r^x) - r^x| \leq \delta.2^i \quad \text{with } i,\ |r^x| \in]2^i, 2^{i+1}], \tag{3}$$

where δ is a constant that depends on the floating-point format.

The sum of the DS structures defining $\hat{r}^x$ is a DS structure d_{r^x}. A first approach to compute the error when rounding this real number is thus to compute the image by the error function of each focal element, and of course add the weights when theses images appear several times:

Definition 4 Suppose that the real value of x is given by the DS structure $d_{r^x} = \{\langle \boldsymbol{x_k^r}, w_k^r \rangle,\ k = 1, \ldots, n\}$. Then for all $k \in [1, n]$, we define i_k such that $max(|\underline{\boldsymbol{x_k^r}}|, |\overline{\boldsymbol{x_k^r}}|) \in]2^{i_k}, 2^{i_k+1}]$. The rounding error of x is defined by the DS

$$d_{e^x} = \rho(\{\langle \boldsymbol{x_k^e}, w_k^e \rangle,\ k = 1, \ldots, n\}),$$

where ρ is a reduction operator that bounds the number of focal elements, and

$$\boldsymbol{x_k^e} = \begin{cases} [-\delta 2^{i_k}, \delta 2^{i_k}] & \text{if } f(\underline{\boldsymbol{x_k^r}}) \neq f(\overline{\boldsymbol{x_k^r}}) \\ [e(\underline{\boldsymbol{x_k^r}}), e(\overline{\boldsymbol{x_k^r}})] & \text{otherwise} \end{cases}$$

This definition for the error can be explained by the fact that if all real values in the focal element are rounded to the same floating-point number, the error will be monotone in the interval, so the error can be bounded by $[e(\underline{\boldsymbol{x_k^r}}), e(\overline{\boldsymbol{x_k^r}})]$. When that is not the case, we can only use Eq. (3), which will result in embedded focal elements $[-\delta 2^{i_k}, \delta 2^{i_k}]$ for the errors. Of course, a less accurate computation would be to consider only the $[-\delta 2^{i_k}, \delta 2^{i_k}]$ focal elements.

Example 5 Consider r^x given in range [200000000, 200000032] (for instance $\hat{r}^x = 2000000016 + 16\varepsilon_1$). The real values 200000000, 200000016 and 200000032 are exactly represented as simple floating-point numbers, the other values in this range must be rounded. Let's say the DS structure defining r^x is $d_{r^x} = \{\langle$[200000000, 200000007], 1/8$\rangle, \langle$[200000009, 200000016], 2/8$\rangle$, $\langle$[2000000016, 200000023], 2/8$\rangle$, $\langle$[200000025, 200000032], 3/8$\rangle\}$ We have here a case where the images of the focal elements by the rounding error function (see Fig. 3) can be computed exactly: they are $[-7, 0]$, $[0, 7]$, $[-7, 0]$, $[0, 7]$, which, after reduction, gives for the error $d_{e^x} = \{\langle[-7, 0], 3/8\rangle, \langle[0, 7], 5/8\rangle\}$.

Example 6 Consider a given integer n and the sum y_p of p independent variables $x_i,\ i = 1, \ldots, p$, each of them defined in $[-1, 1]$ by the DS with $2n$ non-overlapping

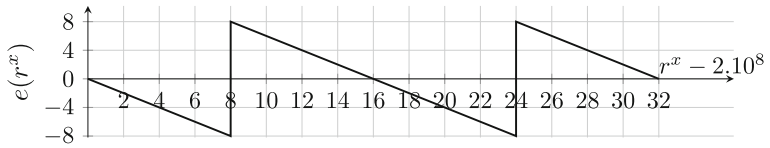


Fig. 3 Rounding error of $r^x \in [2.10^8, 2.10^8 + 32]$ to a simple float (Example 5)

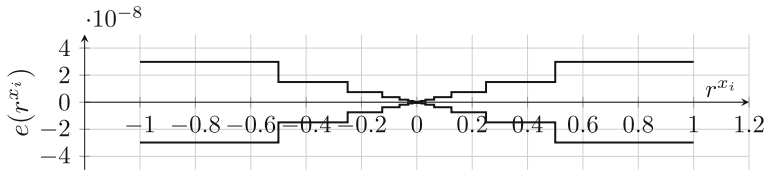
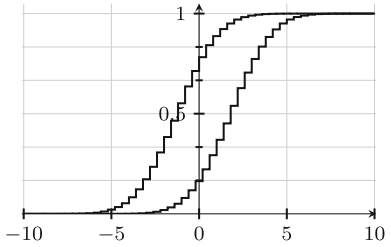


Fig. 4 Envelop on the rounding error of $r^{x_i} \in [-1, 1]$ to a simple float (Example 6)

Fig. 5 P-box for y_p



elements of width $\frac{1}{n}$ and same weight. These variables are first rounded as floating-point values before being added. The envelope of the rounding error on a variable x_i is given in Fig. 4. Consider for instance $n = 5$. The focal elements on the real value are materialized on this Figure by the grid. The image by the rounding error function of the focal elements of r_{x_i} give as DS for the error

$$d_{e^{x_i}} = \left\{ \left\langle \left[-\frac{\delta}{2}, \frac{\delta}{2} \right] 3/5 \right\rangle, \left\langle \left[-\frac{\delta}{4}, \frac{\delta}{4} \right] 1/5 \right\rangle, \left\langle \left[-\frac{\delta}{8}, \frac{\delta}{8} \right] 1/5 \right\rangle \right\}.$$

We compute the partial results $y_k = \sum_{0 \le i < k} x_i$ for k from 1 to p, and a new rounding error $e(r^{y_k})$ is associated with each of these partial results. Finally, the rounding result on y_p is thus $e_p^y = \sum_{0 \le i < p} e(r^{x_i}) + \sum_{1 \le i \le p} e(r^{y_k})$, where the DS for $e(r^{x_i})$ are all independent between one another, and $e(r^{y_k})$ has unknown dependency to $e(r^{x_j})$ and $e(r^{y_j})$ for all $j \le k$.

So it is interesting to consider not only independence or dependency, but a generalized notion of a dependency matrix between variables or noise symbols.

We present in Figs. 5 and 6 the cumulative distribution functions of the DS obtained for $n = 10$ and $p = 10$.

We see in Fig. 5 that we bound indeed a quasi-Gaussian distribution (sum of independent uniform laws). Figure 6 is compatible with the stochastic modeling of errors as popularized in the CESTAC method [19]. The advantage here is that we have guaranteed lower and upper probabilities for rounding errors, whereas the CESTAC

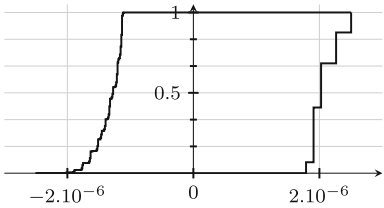


Fig. 6 P-box for the total error on y_p

Table 1 Execution time (t) and dispersion (d) of the Butterworth filter when fixing the discretization parameters (K and p)

n:	10	20	40	50	75	100
t (s)	0.016	0.16	0.023	0.032	0.053	0.084
$t+\gamma$(s)	0.511	1.127	2.295	2.907	4.322	5.712
t (s)	*29.840*	*56.091*	*161.054*	*183.089*	*187.868*	*204.558*
d	1.21809	0.503958	0.493573	0.490891	0.487261	0.487556
d	*11,185*	1.82×10^{7}	$\geq10^{13}$	$\geq10^{17}$	$\geq10^{24}$	$\geq10^{32}$

Values in italics show results using DS arithmetic

method only gives qualitative information about the computation. Here we can read from Fig. 6 that the probability that the error is within $[-1.2.10^{-6}, 1.2.10^{-6}]$ is $\frac{1}{2}$ while the probability that the error is in absolute value greater than 2.10^{-6} (which is about the maximal deterministic error one can get) is almost zero. An experiment with CADNA free allows us to estimate that the first 5 to 7 digits are correct in the calculation of the sum, and that the error is less, in absolute value, than 10^{-6} with a probability of a bit more than $\frac{1}{3}$, which is compatible with the results we get here.

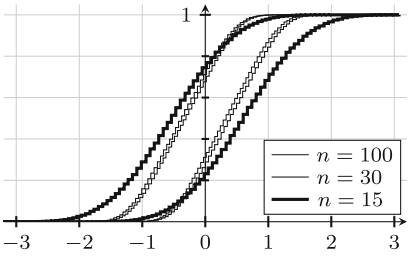
5 Experimental results

We developed a C++ library that implements this new arithmetic. We present here some experiments conducted on a laptop with 4 Gb of RAM and two 2 GHz processors. As in [11], we measure the accuracy of a DS by the sum of all areas (width times weight) of its focal elements. We call this measure the *dispersion*.

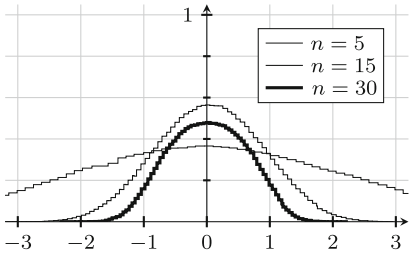
We start with the Butterworth filter of Sect. 1, with uniform law for the inputs. In all our experiments, variable n represents the number of filter iterations, variable K the input discretization, and variable p the output discretization for each operation, i.e. the maximal number of focal elements allowed in DS structures.

Influence of the number of iterations. We set the input discretization to $K = 10$, the output discretization to $p = 400$ and increased n. Results are given in Table 1: the line labeled by t is the time to compute the resulting affine form, and the line labeled by $t + \gamma$ is the time to compute the affine form and convert it into a DS structure (Definition 2). We see that computing the affine form is almost immediate. What takes most of the time is the addition of DS structures by the concretization function.

The dispersion of the output DS structure with affine arithmetic (line labeled by d) decreases as n increases, which is a nice result: the output converges towards a normal



(a) P-boxes for various number of iterations.



(b) Over-approximation of the density function, for various number of iterations.

Fig. 7 P-boxes and density functions for the filter output using probabilistic affine forms

Table 2 Execution time (t) and dispersion (d) of the Butterworth filter when fixing the number of iterations ($n = 30$)

K:	20	30	50	75	100	125
$t + \gamma$(s)	0.182	0.525	3.998	19.849	74.950	149.885
t(s)	*5.448*	*71.800*	*T.O.*	*T.O.*	*T.O.*	*T.O.*
d	1.01035	0.673648	0.404913	0.268924	0.201146	0.160113
d	2.79×10^{10}	2.58×10^{10}	*T.O.*	*T.O.*	*T.O.*	*T.O.*

Values in italics represent the results using DS arithmetic

distribution. On Fig. 7, we show the p-boxes and the corresponding upper bounds on the density functions computed for various numbers of iterations: we clearly see the shape of a normal distribution, as was expected from the Matlab simulation. Table 1 also shows the results obtained using DS arithmetic only (rows shaded in gray). We see that DS arithmetic is very time consuming and that the dispersion is extremely large.

Influence of the input discretization Next, we set the number of iterations ($n = 30$), increase the input discretization K and set $p = 2 \times K$. Results are given in Table 2. As expected, the accuracy of the output DS strongly depends on the discretization of the input DS. We conducted the same tests using DS arithmetic only (rows shaded in gray). This time, we were not able to obtain results in less than 5 min for $K \geq 50$, this is what we denote as *time-out* (T.O.). Using probabilistic affine forms reduces the computation time while keeping the p-box tight. We here fully benefit from the fact that a probabilistic affine form keeps tracks of the linear dependencies between variables.

Finally, we evaluate the function $f(x, y) = x * y - x$ where x is a uniform random variable of support $[2, 3]$ and y is uniform in $[0.5, 1.5]$, x and y being independent. We set the output discretization to $p = 100$ and give in Table 3 the results for different input discretization steps K. Execution times using affine arithmetic and DS arithmetic are more or less equivalent (affine arithmetic being slightly more efficient), as we perform in both cases the same number of operations between DS structures. The precision, however, is much better for the affine arithmetic: the dispersion is divided

Table 3 Execution time (t) and dispersion (d) of $x * y - x$ (in italics: with DS arithmetic)

K:	10	20	30	50	75	100
$t + \gamma$(s)	0.237	1.170	3.693	25.442	95.128	154.059
t(s)	*0.341*	*1.671*	*7.496*	*32.481*	*91.013*	*183.365*
d	0.6532	0.572462	0.53449	0.509894	0.502003	0.49782
d	*1.3879*	*1.24479*	*1.18648*	*1.146771*	*1.12277*	*1.10637*

by 2. Note that the execution time strongly increases with the number of input focal elements, because we need to perform arithmetic with unknown dependency, which is very costly.

Note that all experiments presented are with our C++ library. More experiments using existing DS tools were conducted, but these proved less efficient: some results, along with the matlab script files used, are available at http://www.lix.polytechnique.fr/~bouissou/codes_computing/.

6 Conclusion and future work

In this article, we introduced a new model that mixes interval and probabilistic uncertainties. Other extensions of p-box or DS structures were proposed [2,17,18]: these works are orthogonal to ours as they only differentiate between independent variables and variables with unknown dependency. We believe that an aproach like ours could be used to improve these methods, as well as the method used in RiskCalc for example [4,20].

We aim at generalizing this work in different directions. First of all, we wish to apply this to the static analysis of programs. This means we have to define set-theoretic operations such as union and intersection. They have been introduced in particular in [9] for the underlying affine sets, but have to be generalized to our framework.

Another direction is to maintain a better abstraction of the dependencies between variables, during computation. In the current work, we decomposed the dependency relationships between variables on a basis of independent noise symbols (ε_i) and noise symbols with unknown dependency (η_j). This is a particular abstraction of some form of a correlation matrix between the noise symbols. We propose, in future work, to extend our affine arithmetic based calculus using more precise information about the dependencies, or *the copula* [13], between each pair of noise symbols. Other possibilities would be to use known contraints on probability distributions, such as in [21]. Note that the probabilistic affine forms we have introduced should be linked with particular Dempster–Shafer structures on $\mathbb{R}^p$, whose focal elements are zonotopes (particular central symmetric polytopes), parametrized by affine forms. These might generalize to higher-order Taylor models [12] instead of order one models such as zonotopes.

Acknowledgments We thank the anonymous reviewers for their helpful comments and improvement suggestions. We also want to thank them for pointing interesting references, especially about PERT dia-

grams [15] which seems to be a promising application of our framework. We also want to thank Scott Ferson for his valuable help on the comparison with the RiskCalc tool.

References

1. Berleant D, Goodman-Strauss C (1998) Bounding the results of arithmetic operations on random variables of unknown dependency using intervals. Reliab Comput 4(2):147–165
2. Busaba J, Suwan S, Kosheleva O (2010) A faster algorithm for computing the sum of p-boxes. J Uncertain Syst 4(4):244–249
3. Comba JLD, Stolfi J (1993) Affine arithmetic and its applications to computer graphics. SEBGRAPI'93
4. Ferson S (2002) RAMAS Risk Calc 4.0 Software: risk assessment with uncertain numbers. Lewis Publishers, Boca Raton
5. Ferson S, Kreinovich V, Ginzburg L, Myers D, Sentz K (2003) Constructing probability boxes and Dempster–Shafer structures. Tech. Rep. SAND2002-4015
6. Ferson S, Nelsen R, Hajagos J, Berleant D, Zhang J, Tucker W, Ginzburg L, Oberkampf W (2004) Dependence in probabilistic modelling, Dempster–Shafer theory and probability bounds analysis. Tech. rep., Sandia National Laboratories
7. Frank MJ, Nelsen RB, Schweizer B (1987) Best-possible bounds for the distribution of a sum, a problem of Kolmogorov. Prob Theory Rel Fields 74:199–211. doi:10.1007/BF00569989
8. Ghorbal K, Goubault E, Putot S (2010) A logical product approach to zonotope intersection. In: CAV, LNCS, vol 6174
9. Goubault E, Putot S (2009) A zonotopic framework for functional abstractions. In: CoRR. abs/0910.1763
10. Goubault E, Putot S (2011) Static analysis of finite precision computations. In: VMCAI, LNCS, vol 6538, pp 232–247
11. Limbourg P, Savi R, Petersen J, Kochs HD (2007) Fault tree analysis in an early design stage using the Dempster–Shafer theory of evidence. In: ESREL 2007, pp 713–722
12. Makino K, Berz M (2003) Taylor models and other validated functional inclusion methods. Int J Pure Appl Math 4(4):379–456
13. Nelsen R (1999) An introduction to copulas. In: Lecture notes in statistics. Springer, Berlin
14. Regan HM, Ferson S, Berleant D (2004) Equivalence of methods for uncertainty propagation of real-valued random variables. Int J Approx Reason 36(1):1–30
15. Sanders WH, Meyer JF (2000) Stochastic activity networks: Formal definitions and concepts. In: European Educational Forum, pp 315–343
16. Shafer G (1976) A mathematical theory of evidence. Princeton University Press, Princeton
17. Sun J, Huang Y, Li J, Wang JM (2008) Chebyshev affine arithmetic based parametric yield prediction under limited descriptions of uncertainty. In: ASP-DAC '08. IEEE Computer Society Press, Los Angeles, pp 531–536
18. Terejanu G, Singla P, Singh T, Scott PD (2010) Approximate interval method for epistemic uncertainty propagation using polynomial chaos and evidence theory. In: American Control Conference
19. Vignes J (1993) A stochastic arithmetic for reliable scientific computation. Math Comput Simul 35(3):233–261
20. Williamson RC, Downs T (1990) Probabilistic arithmetic I: numerical methods for calculating convolutions and dependency bounds. J Approx Reason 4(2):89–158
21. Zhang J, Berleant D (2005) Arithmetic on random variables: squeezing the envelopes with new joint distribution constraints. In: ISIPTA, pp 416–422