Error Analysis of Sum-Product Algorithms under Stochastic Rounding

Pablo de Oliveira Castro, El-Mehdi El Arar, Eric Petit, Devan Sohier

ERROR ANALYSIS OF SUM-PRODUCT ALGORITHMS UNDER STOCHASTIC ROUNDING*

PABLO DE OLIVEIRA CASTRO†, EL-MEHDI EL ARAR‡, ERIC PETIT§, AND DEVAN SOHIER†

Abstract. The quality of numerical computations can be measured through their forward error, for which finding good error bounds is challenging in general. For several algorithms and using stochastic rounding (SR), probabilistic analysis has been shown to be an effective alternative for obtaining tight error bounds. This analysis considers the distribution of errors and evaluates the algorithm's performance on average. Using martingales and the Azuma-Hoeffding inequality, it provides error bounds that are valid with a certain probability and in $\mathcal{O}(\sqrt{n}u)$ instead of deterministic worst-case bounds in $\mathcal{O}(nu)$, where n is the number of operations and u is the unit roundoff. In this paper, we present a general method that automatically constructs a martingale for any computation scheme with multi-linear errors based on additions, subtractions, and multiplications. We apply this generalization to algorithms previously studied with SR, such as pairwise summation and the Horner algorithm, and prove equivalent results. We also analyze a previously unstudied algorithm, Karatsuba polynomial multiplication, which illustrates that the method can handle reused intermediate computations.

Key words. Stochastic rounding, Martingales, Rounding error analysis, Floating-point arithmetic, Computation DAG, Karatsuba multiplication

MSC codes. 65G50, 65F35, 60G44

1. Introduction.

Stochastic Rounding (SR) is a rounding mode for floating-point numbers in which the rounding direction is chosen at random, inversely proportionally to the relative distance to the nearest representable values. SR is an alternative to the more common deterministic rounding that has drawn attention in recent years [16], in particular due to its resilience to stagnation [18, 17]; the phenomenon in which the accumulator in long summations become so big that the remaining individual terms to be summed become negligible with respect to the precision in use, even if their exact sum is not. Indeed, for summations, RN has worst-case error bounds proportional to the number of floating-point operations n. With high probability, SR has error bounds [2] proportional to $\sqrt{n}$. SR is more robust than RN because the randomness removes the bias in the accumulation of errors.

For example, during parameter updates in deep learning, SR avoids stagnation, particularly when using low-precision formats for computations or storage [11]. In *gradient descent*, when computing the minimum of a function using RN-binary16 precision [18, 17], it has been observed that the gradient can approach zero too quickly, causing the update to be lost due to limited precision. SR mitigates this issue by maintaining some accuracy on average, preventing stagnation in such scenarios.

Until now, computing SR probabilistic error bounds has been done case-by-case with algorithm-dependent proofs. Available proofs in the literature fall into two main schemes. A first proof scheme [1, 13, 8, 12] models the algorithm's error as a stochastic process Ψ_k, shows that it is a martingale, and computes an error bound with Azuma-

*Version of November 18, 2024.

Funding: This work was funded by the HOLIGRAIL (ANR-23-PEIA-0010), the INTERFLOP (ANR-20-CE46-0009), and FPT-4 (ANR-24-CE46-7572) projects.

†Université Paris-Saclay, UVSQ, Li-PaRAD, Saint-Quentin en Yvelines, France (pablo.oliveira@uvsq.fr, devan.sohier@uvsq.fr).

‡Université de Rennes, Inria, IRISA, Rennes, France (el-mehdi.el-arar@inria.fr).

§Intel Corp , Portland, USA (eric.petit@intel.com).

Hoeffding concentration inequality. A second scheme [9], bounds the variance of Ψ_k and applies Chebyshev concentration inequality.

This paper generalizes the computation of SR error bounds to all algorithms that can be modeled as a computation DAG comprised of sums, subtractions, and multiplications, as long as no multiplication node has two children sharing a common ancestor that is not an input (this is, in particular, true of all computation trees).

Section 3 proves through structural induction that the errors in such computation DAGs form a martingale. It also gives a systematic recursive formulation to bound the martingale increments, allowing the use of Azuma-Hoeffding inequality to compute a probabilistic error bound of the whole computation DAG.

In Section 4, we apply the method to generalize previous results on pairwise summation [12] and Horner's polynomial evaluation [9]. Moreover, to the best of our knowledge, we are the first to investigate Karatsuba polynomial multiplication under SR. We demonstrate the applicability of the generalization proved in Section 3 to bound the forward error of this algorithm under SR.

With SR, the algorithmic errors are captured through a stochastic process. We propose to use an important result from martingale theory, the Doob-Meyer decomposition [5], to decompose the error stochastic process into a martingale and a predictable drift. We show that the computation trees analyzed in previous sections always have a zero drift in such decomposition. The paper closes with a discussion of possible directions to analyze algorithms with a non-zero drift term.

2. Preliminaries. Throughout this paper, $\widehat{x} = x(1+\delta)$ is the approximation of the real number x under stochastic rounding, with $|\delta| \leq u$ and u is the unit roundoff. If x is representable, $\widehat{x} = x$ and $\delta = 0$. For a non-representable $x \in \mathbb{R}$, denote $p(x) = \frac{x - \lfloor\!\lfloor x \rfloor\!\rfloor}{\lceil\!\lceil x \rceil\!\rceil - \lfloor\!\lfloor x \rfloor\!\rfloor}$, where $\lceil\!\lceil x \rceil\!\rceil$ is the smallest floating-point number upper than x, and $\lfloor\!\lfloor x \rfloor\!\rfloor$ is the greatest floating-point number lower than x. Note that if x is representable, $x = \lceil\!\lceil x \rceil\!\rceil = \lfloor\!\lfloor x \rfloor\!\rfloor$. We consider the following stochastic rounding mode, called SR-nearness:

$$\widehat{x} = \begin{cases} \lceil\!\lceil x \rceil\!\rceil & \text{with probability } p(x), \\ \lfloor\!\lfloor x \rfloor\!\rfloor & \text{with probability } 1 - p(x). \end{cases}$$

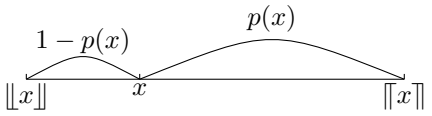


Fig. 1: **SR-nearness**.

The rounding SR-nearness mode is unbiased (which does not mean that a sequence of operations using SR is necessarily unbiased; for instance, squaring an unbiased error leads to a bias due to the square term that corresponds to a variance):

$$\begin{aligned} E(\widehat{x}) &= p(x)\lceil\!\lceil x \rceil\!\rceil + (1 - p(x))\lfloor\!\lfloor x \rfloor\!\rfloor \\ &= p(x)(\lceil\!\lceil x \rceil\!\rceil - \lfloor\!\lfloor x \rfloor\!\rfloor) + \lfloor\!\lfloor x \rfloor\!\rfloor = x. \end{aligned}$$

The following lemma has been proven in [1, lem 5.2] and shows that rounding errors under SR-nearness are mean independent.

LEMMA 2.1. *Let a and b be the result of $k-1$ scalar operations and $\delta_1, \ldots, \delta_{k-1}$ be the rounding errors obtained using SR-nearness. Consider $c \leftarrow a \operatorname{op} b$ for $\operatorname{op} \in \{+, -, \times, /\}$, and δ_k the error of the k^{th} operation, that is to say, $\hat{c} = (a \operatorname{op} b)(1+\delta_k)$.*

The δ_k are random variables with mean zero and $(\delta_1, \delta_2, \ldots)$ is mean independent, i.e., $\forall k \geq 2, \mathbb{E}[\delta_k \mid \delta_1, \ldots, \delta_{k-1}] = \mathbb{E}(\delta_k)$.

DEFINITION 1. *A sequence of random variables $M_1, \cdots, M_n$ is a martingale with respect to the sequence $X_1, \cdots, X_n$ if, for all k,*

- *M_k is a function of $X_1, \cdots, X_k$,*
- *$\mathbb{E}(|M_k|) < \infty$, and*
- *$\mathbb{E}[M_k / X_1, \cdots, X_{k-1}] = M_{k-1}$.*

LEMMA 2.2. *(Azuma-Hoeffding inequality). Let $M_0, \cdots, M_n$ be a martingale with respect to a sequence $X_1, \cdots, X_n$. We assume that there exist $a_k < b_k$ such that $a_k \leq M_k - M_{k-1} \leq b_k$ for $k = 1 : n$. Then, for any $A > 0$*

$$\mathbb{P}(|M_n - M_0| \geq A) \leq 2 \exp\left(-\frac{2A^2}{\sum_{k=1}^{n}(b_k - a_k)^2}\right).$$

In the particular case $a_k = -b_k$ and $\lambda = 2\exp\left(-\frac{A^2}{2\sum_{k=1}^{n} b_k^2}\right)$ we have

$$\mathbb{P}\left(|M_n - M_0| \leq \sqrt{\sum_{k=1}^{n} b_k^2}\sqrt{2\ln(2/\lambda)}\right) \geq 1 - \lambda,$$

where $0 < \lambda < 1$.

Remark 2.3. In Lemma 2.2, if all b_k are constant with value b,

$$\sqrt{\sum_{k=1}^{n} b_k^2} = \sqrt{\sum_{k=1}^{n} b^2} = |b|\sqrt{n}.$$

It has been shown [8, 9, 10, 13, 12, 1] that the mean independence property is sufficient to improve the error analysis of algorithms with SR-nearness. It leads to obtaining a martingale (Definition 1), which is a sequence of random variables such that the expected value of the next value in the sequence, given all the past values, is equal to the current value. Using Azuma-Hoeffding inequality [15, p 303], allows to obtain probabilistic bounds on the error in $\mathcal{O}(\sqrt{n}u)$. For further details, we refer to [6, chap 4].

3. Errors in sum-product computation graphs. In this section, by induction, and for any computation, we build a martingale the last term of which is the rounding error of the computation. This construction gives the length of this martingale, as well as a condition number based on a deterministic bound on the martingale steps. Together, these quantities allow to apply Azuma-Hoeffding inequality, or to compute the variance of the error, and thus to probabilistically bound the rounding error of the computation.

This martingale generalizes the ones found for the recursive summation [1], the dot product [13], the pairwise summation [12, 10], and the Horner's polynomial evaluation [8]. This construction applies to any numerical scheme based on additions and products, in which no two variables sharing a rounding error are multiplied. Seeing the computation as a DAG, the parents of a multiplication node cannot share a common ancestor (except for inputs, which are not affected by an error). In the case a common ancestor exists for multiplication nodes, then a bias appears (the expectation of the squared error is not zero), which this method cannot account for. We propose in the last section a method to deal with such biases.

The construction differs according to its last operation. For a sum, the martingale is basically the weighted sum of the two martingales associated to the summands, with one additional term for the last error: the length of the resulting martingale is the length of the longest of the two plus one, and the bound on the step is the weighted mean of the two bounds with the values of the summands as coefficients.

For a product, the martingale is built by ordering the two martingales of the multiplied terms, and adding one term accounting for the last error. This ordering requires that no individual rounding error is shared in both terms, which forbids that any part of the two terms depend on the same computation, as previously stated. The length of the resulting martingale is the sum of the lengths of the two plus one, and the bound on the step is the product of the two bounds associated to the operands.

In Figure 2, we consider an algorithm in which `z` is the return value, and the last operation is $\mathtt{z} \leftarrow \mathtt{x}\,\mathrm{op}\,\mathtt{y}$:

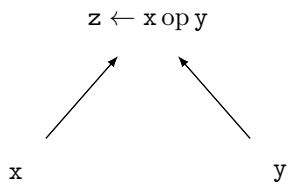


Fig. 2: Last operation in the computation of a variable z, $\mathrm{op} \in \{+, -, \times\}$.

3.1. Base case: `z` is an input. The base case is straightforward. Since we assume that inputs are exact (void computations), the error is 0, and it can be seen the last term of the trivial martingale consisting of the empty sequence. The length of this martingale is 0, and the associated condition number is 1.

3.2. Addition. Suppose that the last operation in the computation of the variable z is an addition, i.e, $\mathtt{z} \leftarrow \mathtt{x}+\mathtt{y}$. Consider the relative errors Φ, X and Ψ associated respectively to `x`, `y`, and `z`. Note x, y, and z their respective exact values, and $\hat{x}$, $\hat{y}$ and $\hat{z}$ their computed values. Therefore:

$$\begin{cases} \hat{x} = x(1+\Phi), \\ \hat{y} = y(1+\mathrm{X}), \\ \hat{z} = z(1+\Psi). \end{cases} \tag{3.1}$$

We have $z = x + y$, then, there exists δ such that $\hat{z} = (\hat{x} + \hat{y})(1+\delta)$. Hence,

$$\begin{aligned} \Psi &= \frac{\hat{z} - z}{z} \\ &= \frac{\hat{x} + \hat{y}}{x+y}(1+\delta) - 1 \\ &= \left(1 + \frac{x}{x+y}\Phi + \frac{y}{x+y}\mathrm{X}\right)(1+\delta) - 1. \end{aligned}$$

Suppose by induction that there exist constants $K_x \geq 1$ (bounding the condition number in the computation of x) and $K_y \geq 1$ (bounding the condition number in the

computation of y), and martingales $(\Phi_i)_{i=0}^{k-1}$ and $(\mathrm{X}_i)_{i=0}^{l-1}$ with their i^{th} step $|\Phi_i - \Phi_{i-1}|$ and $|\mathrm{X}_i - \mathrm{X}_{i-1}|$ bounded respectively by $K_x u(1+u)^{i-1}$ and $K_y u(1+u)^{i-1}$, such that $\Phi_0 = 0$, $\mathrm{X}_0 = 0$, $\Phi = \Phi_{k-1}$ and $\mathrm{X} = \mathrm{X}_{l-1}$. When one of $\hat{x}$ or $\hat{y}$ is exact, as mentioned in Subsection 3.1, we assume that the length of the martingale is 0 and the condition number is 1.

LEMMA 3.1. *Let $m = \max\{k, l\} + 1$. The stochastic process $(\Psi_i)_{i=0}^{m-1}$ such that $\Psi_{m-1} = \Psi$, and for all $0 \leq i < m-1$,*

$$\Psi_i = \frac{x}{x+y}\Phi_{\min\{i,k\}} + \frac{y}{x+y}\mathrm{X}_{\min\{i,l\}},$$

forms a martingale.

Proof. Without loss of generality, let us assume that $k \leq l$. Then, $m = l + 1$ and

$$\begin{cases} \Psi_i &= \frac{x}{x+y}\Phi_i + \frac{y}{x+y}\mathrm{X}_i \text{ for all } 0 \leq i \leq k-2 \\ \Psi_i &= \frac{x}{x+y}\Phi + \frac{y}{x+y}\mathrm{X}_i \text{ for all } k-1 \leq i \leq l-1. \end{cases}$$

Note that $(\Phi_i)_{i=0}^{m-2}$ with $\Phi_i = \Phi$ for all $k-1 \leq i \leq m-2$ and $(\mathrm{X}_i)_{i=0}^{m-2}$ are martingales by induction hypothesis. Since the martingale set is a vector space, as a linear combination of them, $(\Psi_i)_{i=0}^{m-2}$ is a martingale. Moreover, by mean independence (Lemma 2.1) of δ from Φ and X we have

$$\begin{aligned} E[\Psi_{m-1}/\Psi_{m-2}] &= E\left[\left(1 + \frac{x}{x+y}\Phi + \frac{y}{x+y}\mathrm{X}\right)(1+\delta) - 1/\Psi_{m-2}\right] \\ &= \left(1 + \frac{x}{x+y}\Phi + \frac{y}{x+y}\mathrm{X}\right) E[(1+\delta)/\Psi_{m-2}] - 1 \\ &= \Psi_{m-2}. \end{aligned}$$

∎

Thus, $(\Psi)_{i=0}^{m-1}$ is a martingale and $\Psi_{m-1} = \Psi$.

In this lemma, we have built a martingale by induction when the last operation is an addition. In order to use Azuma-Hoeffding inequality (Lemma 2.2), we have to bound martingale increments.

LEMMA 3.2. *Let $K_z = \frac{|x|}{|x+y|}K_x + \frac{|y|}{|x+y|}K_y$. The martingale $(\Psi_i)_{i=0}^{m-1}$ satisfies*

$$|\Psi_i - \Psi_{i-1}| \leq u C_i,$$

where $C_i = K_z(1+u)^{i-1}$ for all $1 \leq i \leq m-1$.

Proof. For all $0 \leq i < m-1$ by definition of Ψ_i, we have

$$\Psi_i - \Psi_{i-1} = \frac{x}{x+y}(\Phi_{\min\{i,k-1\}} - \Phi_{\min\{i-1,k-1\}}) + \frac{y}{x+y}(\mathrm{X}_{\min\{i,l-1\}} - \mathrm{X}_{\min\{i-1,l-1\}}).$$

Then, by induction hypothesis we get

$$\begin{aligned} |\Psi_i - \Psi_{i-1}| \leq& \left|\frac{x}{x+y}\right| \left|\Phi_{\min\{i,k-1\}} - \Phi_{\min\{i-1,k-1\}}\right| \\ &+ \left|\frac{y}{x+y}\right| \left|\mathrm{X}_{\min\{i,l-1\}} - \mathrm{X}_{\min\{i-1,l-1\}}\right| \\ \leq& \frac{|x|}{|x+y|}K_x u(1+u)^{\min\{i-1,k-1\}} + \frac{|y|}{|x+y|}K_y u(1+u)^{\min\{i-1,l-1\}} \\ \leq& K_z u(1+u)^{i-1}. \end{aligned}$$

Moreover, for $i = m-1$

$$\begin{aligned}|\Psi_{m-1} - \Psi_{m-2}| &= \left|\left(1 + \frac{x}{x+y}\Phi + \frac{y}{x+y}\mathrm{X}\right)\delta\right| \\ &\leq u\left|\frac{x}{x+y}(\Phi+1) + \frac{y}{x+y}(\mathrm{X}+1)\right|.\end{aligned}$$

Since $\Phi_0 = 0$,

$$\begin{aligned}|\Phi+1| = |\Phi_{k-1}+1| &= \left|1 + \sum_{j=1}^{k-1}(\Phi_j - \Phi_{j-1})\right| \\ &\leq 1 + \sum_{j=1}^{k-1}|\Phi_j - \Phi_{j-1}| \\ &\leq 1 + \sum_{j=1}^{k-1} uK_x(1+u)^{j-1} \\ &= 1 + uK_x\frac{(1+u)^{k-1}-1}{u} \\ &= 1 + K_x(1+u)^{k-1} - K_x \\ &\leq K_x(1+u)^{k-1}.\end{aligned}$$

The same method shows that $|X+1| \leq K_y(1+u)^{l-1}$. It follows that

$$\begin{aligned}|\Psi_{m-1} - \Psi_{m-2}| &\leq u\left(\frac{|x|}{|x+y|}K_x(1+u)^{k-1} + \frac{|y|}{|x+y|}K_y(1+u)^{l-1}\right) \\ &\leq K_z u(1+u)^{m-2}.\end{aligned}$$

∎

Corollary 3.3. *For all $0 < \lambda < 1$, the computed $\hat{z}$ in Equation (3.1) satisfies under SR-nearness*

$$\frac{|\hat{z}-z|}{|z|} \leq K_z\sqrt{u\gamma_{2(m-1)}(u)}\sqrt{\ln(2/\lambda)}, \tag{3.2}$$

with probability at least $1-\lambda$.

Proof. Using Azuma-Hoeffding inequality, we have

$$\frac{|\hat{z}-z|}{|z|} = |\Psi_{m-1}| \leq \sqrt{\sum_{i=1}^{m-1} u^2 C_i^2}\sqrt{2\ln(2/\lambda)},$$

with probability at least $1-\lambda$. Moreover,

$$\begin{aligned}\sum_{i=1}^{m-1} u^2 C_i^2 &= u^2 K_z^2 \sum_{i=1}^{m-1}(1+u)^{2(i-1)} \\ &= u^2 K_z^2 \frac{(1+u)^{2(m-1)}-1}{u^2+2u} \\ &\leq uK_z^2\frac{\gamma_{2(m-1)}(u)}{2}.\end{aligned}$$

Finally, we get

$$\frac{|\hat{z}-z|}{|z|} \leq K_z \sqrt{u\gamma_{2(m-1)}(u)}\sqrt{\ln(2/\lambda)},$$

with probability at least $1-\lambda$. ∎

3.3. Multiplication. Suppose now that the last operation is a multiplication, i.e, $\mathtt{z} \leftarrow \mathtt{x} \times \mathtt{y}$. Consider the relative errors Φ, X, and Ψ associated respectively to x, y, and z. Note x, y, and z their respective exact values, and $\hat{x}$, $\hat{y}$ and $\hat{z}$ their computed values, with

$$\begin{cases} \hat{x} = x(1+\Phi), \\ \hat{y} = y(1+\mathrm{X}), \\ \hat{z} = z(1+\Psi). \end{cases} \tag{3.3}$$

We have $z = x \times y$, then there exists δ such that $\hat{z} = \hat{x} \times \hat{y}(1+\delta)$. Hence,

$$\begin{aligned} \Psi &= \frac{\hat{z}-z}{z} \\ &= \frac{\hat{x}\times\hat{y}}{x\times y}(1+\delta) - 1 \\ &= (1+\Phi)(1+\mathrm{X})(1+\delta) - 1. \end{aligned}$$

We know by induction that Φ and X are the last terms of two martingales. However, the multiplication of two martingales is not necessarily a martingale. Consequently, in contrast to the addition case, we have to decide a scheduling of operations in the construction of the martingale Ψ. All are equivalent and lead to the same final result.

As presented in Lemma 3.4, we assume that in figure 2, the left sub-tree is computed before the right sub-tree, which means that in the computation of x, we assume that we don't have any operation on y. Consider two martingales $(\Phi_i)_{i=0}^{k-1}$ and $(\mathrm{X}_i)_{i=0}^{l-1}$ such that $\Phi_0 = 0$, $\Phi = \Phi_{k-1}$, $\mathrm{X}_0 = 0$, $\mathrm{X} = \mathrm{X}_{l-1}$, and random errors in Φ are different from those of X (thanks to the multi-linearity of errors in the computation of z). The following lemma shows that Ψ is the last term of a martingale built from $(\Phi_i)_{i=0}^{k-1}$ and $(\mathrm{X}_i)_{i=0}^{l-1}$.

LEMMA 3.4. *The stochastic process* $(\Psi_i)_{i=0}^{m-1}$ *such that*

$$\Psi_i = \begin{cases} \Phi_i = (1+\Phi_i)(1+0) - 1 & \text{for all } 0 \leq i \leq k-1 \\ (1+\Phi)(1+\mathrm{X}_{i-k}) - 1 & \text{for all } k \leq i \leq m-2 \\ (1+\Phi)(1+\mathrm{X})(1+\delta) - 1 & \text{for } i = m-1, \end{cases}$$

forms a martingale.

Proof. For all $0 \leq i \leq k-1$, by construction of Ψ, we have $\Psi_i = \Phi_i$. Since $(\Phi_i)_{i=0}^{k-1}$ is a martingale, we have

$$E[\Psi_i/\Psi_{i-1}] = E[\Phi_i/\Psi_{i-1}] = \Phi_{i-1} = \Psi_{i-1}.$$

Moreover, for the k^{th} term we have

$$\begin{aligned} E[\Psi_k/\Psi_{k-1}] &= E[(1+\Phi)(1+\mathrm{X}_1) - 1/\Psi_{k-1}] \\ &= (1+\Phi)E[(1+\mathrm{X}_0)/\Psi_{k-1}] - 1 \\ &= (1+\Phi) - 1 \text{ by Lemma 2.1} \\ &= \Phi. \end{aligned}$$

Since $(\mathrm{X}_{i-k})_{i=k}^{m-2}$ is a martingale, for all $k < i \leq m-2$,

$$\begin{aligned} E[\Psi_i/\Psi_{i-1}] &= E[(1+\Phi)(1+\mathrm{X}_{i-k}) - 1/\Psi_{i-1}] \\ &= (1+\Phi)E[(1+\mathrm{X}_{i-k})/\Psi_{i-1}] - 1 \\ &= (1+\Phi)(1+\mathrm{X}_{i-k-1}) - 1 = \Psi_{i-1}. \end{aligned}$$

By mean independence of δ and Ψ_{m-2}, we get

$$\begin{aligned} E[\Psi_{m-1}/\Psi_{m-2}] &= E[(1+\Phi)(1+\mathrm{X}_{l-1})(1+\delta) - 1/\Psi_{m-2}] \\ &= (1+\Phi)(1+\mathrm{X}_{l-1})E[(1+\delta)/\Psi_{m-2}] - 1 \\ &= \Psi_{m-2}. \end{aligned}$$

∎

In order to use Azuma-Hoeffding inequality (Lemma 2.2), we need to bound the martingale increments. We can show by induction that there exist constants $K_x \geq 1$ (bounding the condition number in the computation of x) and $K_y \geq 1$ (bounding the condition number in the computation of y), such that the i^{th} step $|\Phi_i - \Phi_{i-1}|$ and $|\mathrm{X}_i - \mathrm{X}_{i-1}|$ are bounded respectively by $K_x u(1+u)^{i-1}$ and $K_y u(1+u)^{i-1}$ (because $\mathrm{X}_j = 0$ for all $0 \leq j \leq k-1$).

LEMMA 3.5. *Let $K_z = K_x K_y$. The martingale $(\Psi_i)_{i=0}^{m-1}$ satisfies*

$$|\Psi_i - \Psi_{i-1}| \leq u C_i,$$

where $C_i = K_z(1+u)^{i-1}$ for all $1 \leq i \leq m-1$.

Proof. For all $1 \leq i \leq k-1$, $|\Psi_i - \Psi_{i-1}| = |\Phi_i - \Phi_{i-1}| \leq K_x u(1+u)^{i-1}$. Moreover, for all $k \leq i \leq m-2$,

$$\begin{aligned} |\Psi_i - \Psi_{i-1}| &= |(1+\Phi)(1+\mathrm{X}_i) - (1+\Phi)(1+\mathrm{X}_{i-1})| \\ &= |(1+\Phi)(\mathrm{X}_i - \mathrm{X}_{i-1})| \\ &\leq |1+\Phi_i|\, K_y u(1+u)^{i-k}. \end{aligned}$$

As for the summation case, $|1+\Phi_i| \leq K_x(1+u)^{k-1}$. Then, for all $k \leq i \leq m-2$,

$$|\Psi_i - \Psi_{i-1}| \leq K_x(1+u)^{k-1} K_y u(1+u)^{i-k} = K_z u(1+u)^{i-1}.$$

Finally, we obtain

$$\begin{aligned} |\Psi_{m-1} - \Psi_{m-2}| &= |(1+\Phi)(1+\mathrm{X})\delta| \\ &\leq u K_x(1+u)^{k-1} K_y(1+u)^{m-k-1} \\ &\leq u K_z(1+u)^{m-2}. \end{aligned}$$

∎

COROLLARY 3.6. *For all* $0 < \lambda < 1$*, the computed* $\hat{z}$ *in Equation (3.3) satisfies under SR-nearness*

$$\frac{|\hat{z} - z|}{|z|} \leq K_z \sqrt{u\gamma_{2(m-1)}(u)} \sqrt{\ln(2/\lambda)}, \tag{3.4}$$

with probability at least $1 - \lambda$*.*

Proof. Using Azuma-Hoeffding inequality, we have

$$\frac{|\hat{z} - z|}{|z|} = |\Psi_{m-1}| \leq \sqrt{\sum_{i=1}^{m-1} u^2 C_i^2} \sqrt{2\ln(2/\lambda)},$$

with probability at least $1 - \lambda$. Moreover,

$$\begin{aligned} \sum_{i=1}^{m-1} u^2 C_i^2 &= u^2 K_z^2 \sum_{i=1}^{m-1} (1+u)^{2(i-1)} \\ &= u^2 K_z^2 \frac{(1+u)^{2(m-1)} - 1}{u^2 + 2u} \\ &\leq u K_z^2 \frac{\gamma_{2(m-1)}(u)}{2}. \end{aligned}$$

Finally, we get

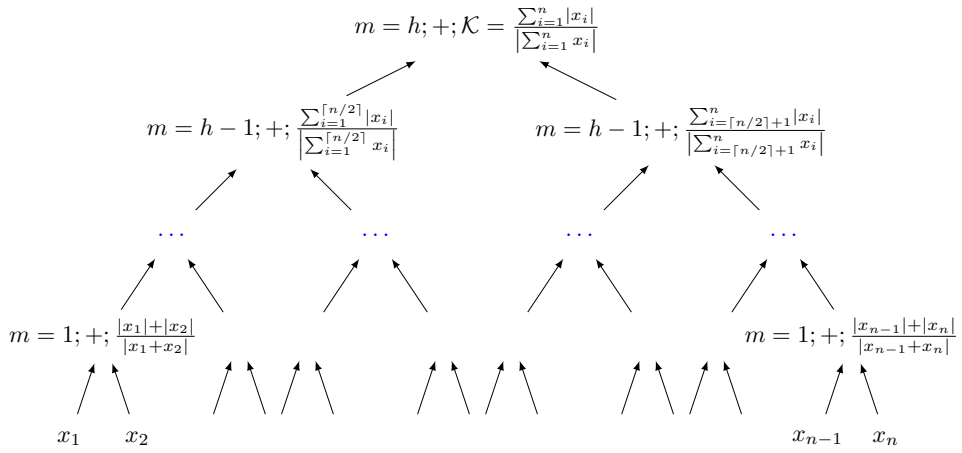
$$\frac{|\hat{z} - z|}{|z|} \leq K_z \sqrt{u\gamma_{2(m-1)}(u)} \sqrt{\ln(2/\lambda)},$$

with probability at least $1 - \lambda$. ∎

Corollaries 3.3 and 3.6 show that the error of any algorithm based on elementary operations $\{+, -, \times\}$ and with multi-linear errors has a probabilistic bound in $\mathcal{O}(\sqrt{n}u)$, where n is the number of operations.

4. Error analysis using the proved generalization. In this section, we apply this generalization to algorithms based on elementary operations $\{+, -, \times\}$ with multi-linear errors. First, we consider the pairwise summation algorithm that involves only additions, and we show how this method computes the martingale's length and the condition number. We obtain the same result proved in [10, 12] for this algorithm. Next, we analyze the Horner algorithm that combines additions and multiplications, which illustrates the effect of multiplication on the martingale length. We obtain the same result proved in [8] for this algorithm. Finally, we examine the Karatsuba algorithm, which demonstrates the flexibility of this method in handling DAGs where, in the case of multiplication, two nodes do not share errors.

4.1. Pairwise summation. We investigate the forward error made by the pairwise summation algorithm under SR. Section 3 demonstrates that the error generated by this algorithm forms a martingale. In the following, we illustrate how the generalization presented in the previous section can be applied to compute the length of this martingale and bound the condition number. We thus use Azuma-Hoeffding inequality to compute a probabilistic bound for the error. For illustrative purposes, let's consider $z = \sum_{i=1}^{n} x_i$ such that ($\lceil n/2 \rceil$ is the smallest integer more than or equal to $n/2$):



At each internal node, we have:

- On the left, m represents the length of the martingale. In this case, $m = 1, 2, \ldots, h$, where h is the height of tree.
- In the middle, we have the elementary operation between the two children. In this case, only additions are considered.
- On the right, we have the current condition number from the leaves up to this node.

Since there are only additions, m is $\max\{m_l, m_r\} + 1$, where m_l and m_r are the martingale lengths at the left and right sub-trees, respectively. We assume that the inputs are exact, so $m = 0$ at each leaf. Consequently, since we add one at each step, at the root, $m = h$, the height of the tree.

Let us compute the condition number of the root. From Lemma 3.2, we have $\mathcal{K} = \frac{|x|}{|x+y|} K_x + \frac{|y|}{|x+y|} K_y$ where $x = \sum_{i=1}^{\lceil n/2 \rceil} x_i$, $y = \sum_{i=\lceil n/2 \rceil+1}^{n} x_i$, and

$$\begin{cases} K_x &= \frac{\sum_{i=1}^{\lceil n/2 \rceil} |x_i|}{\left|\sum_{i=1}^{\lceil n/2 \rceil} x_i\right|} = \frac{\sum_{i=1}^{\lceil n/2 \rceil} |x_i|}{|x|}, \\ K_y &= \frac{\sum_{i=\lceil n/2 \rceil+1}^{n} |x_i|}{\left|\sum_{i=\lceil n/2 \rceil+1}^{n} x_i\right|} = \frac{\sum_{i=\lceil n/2 \rceil+1}^{n} |x_i|}{|y|}. \end{cases}$$

We thus have $x + y = \sum_{i=1}^{n} x_i$ and

$$\begin{aligned} \mathcal{K} &= \frac{|x|}{|x+y|} K_x + \frac{|y|}{|x+y|} K_y \\ &= \frac{1}{|\sum_{i=1}^{n} x_i|} \left(|x| \, K_x + |y| \, K_y \right) \\ &= \frac{1}{|\sum_{i=1}^{n} x_i|} \left(\sum_{i=1}^{\lceil n/2 \rceil} |x_i| + \sum_{i=\lceil n/2 \rceil+1}^{n} |x_i| \right) \\ &= \frac{\sum_{i=1}^{n} |x_i|}{|\sum_{i=1}^{n} x_i|}. \end{aligned}$$

Finally, Corollary 3.3 shows that

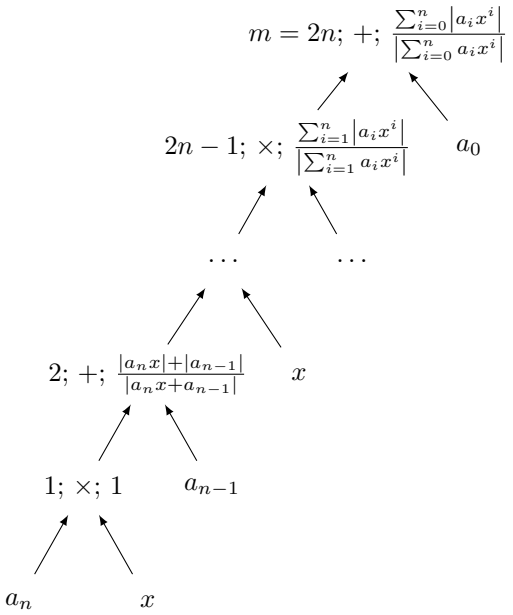
$$\frac{|\hat{z} - z|}{|z|} \leq \mathcal{K} \sqrt{u \gamma_{2h}(u)} \sqrt{\ln(2/\lambda)} = \mathcal{O}(\sqrt{h} u), \tag{4.1}$$

with probability at least $1-\lambda$, $\forall\lambda \in]0;1[$, where $h = \lfloor \log_2(n) \rfloor$. Interestingly, the bound in (4.1) is identical to the bound proved in [10] for the pairwise summation using the AH method.

This proof can easily be adapted to any summation tree, leading to a bound of $\sqrt{h}u$ with high probability, with h the height of the tree.

4.2. Horner algorithm. The previous example only had additions. Let now apply Section 3 to Horner's polynomial evaluation, with both additions and multiplication. Let $P(x) = \sum_{i=0}^{n} a_i x^i$, Horner's algorithm consists in writing this polynomial as

$$P(x) = (((a_n x + a_{n-1})x + a_{n-2})x \ldots + a_1)x + a_0.$$



As in the previous example, internal nodes represent three elements. They show the martingale length on the left, the operation between child nodes in the middle, and the condition number from the leaves to the node on the right.

Let recall that m is $\max\{m_l, m_r\} + 1$ for additions and $m_l + m_r + 1$ for multiplications, where m_l and m_r are the martingale lengths at the left and right sub-trees, respectively. We suppose that $m = 0$ for leaves. In Horner's algorithm, $m_r = 0$, so that both in additions and multiplications, $m = m_l + 1$. Since there are n additions and n multiplications, we have $m = 2n$.

Let us compute the condition number bound. The first operation is a multiplication between a_n and x. According to Lemma 3.5, the condition number is 1. The second operation is an addition between $a_n x$ and a_{n-1}. According to Lemma 3.2, the condition number is $\frac{|a_n x|+|a_{n-1}|}{|a_n x+a_{n-1}|}$. For the root, we have:

$$\begin{aligned}\mathcal{K} &= \frac{\left|\sum_{i=1}^{n} a_i x^i\right|}{|\sum_{i=1}^{n} a_i x^i + a_0|} \frac{\sum_{i=1}^{n} \left|a_i x^i\right|}{|\sum_{i=1}^{n} a_i x^i|} + \frac{|a_0|}{|\sum_{i=1}^{n} a_i x^i + a_0|} \\ &= \frac{\sum_{i=0}^{n} \left|a_i x^i\right|}{|\sum_{i=0}^{n} a_i x^i|}.\end{aligned}$$

Note that the condition number remains the same in the case of multiplication by an input. Finally, Corollary 3.3 and Corollary 3.6 show

$$\frac{\left|\hat{P}(x) - P(x)\right|}{|P(x)|} \leq \mathcal{K}\sqrt{u\gamma_{4n}(u)}\sqrt{\ln(2/\lambda)} = \mathcal{O}(\sqrt{n}u), \tag{4.2}$$

with probability at least $1 - \lambda$. Interestingly, the bound in (4.2) is identical to the bound proved in [8, thm IV.2] for the Horner algorithm.

4.3. Karatsuba polynomial multiplication. Karatsuba multiplication [14] is a divide-and-conquer algorithm that reduces the number of multiplications in the product of two polynomials[1]. There are different variants; here, we consider the substractive variant.

Let us consider two polynomials A and B of degree $2^n - 1$.

- If $n = 0$, the Karatsuba product of A and B reduces to a scalar multiplication, $K(A, B) = a_0.b_0$.
- If $n \geq 1$, we write $A = A_h.X^{2^{n-1}} + A_l$ and $B = B_h.X^{2^{n-1}} + B_l$ where A_h and B_h capture the high order coefficients and A_l and B_l capture the low order coefficients of A and B respectively. Then the product of A and B is

$$K(A, B) = P_2.X^{2^n} + (P_0 + P_1 + P_2).X^{2^{n-1}} + P_0$$

  where $P_0 = K(A_l, B_l)$, $P_2 = K(A_h, B_h)$ and $P_1 = K(A_h - A_l, B_l - B_h)$.

We can note that this recursive step uses only three polynomial multiplications instead of four in a recursive formulation of the classical multiplication algorithm, leading to a complexity of $O(n^{\log_2(3)})$ instead of $O(n^2)$.

The following result allows to apply the results proven in the previous section. Figure 3 illustrates on a product of polynomials of degree 3 how one term of all the multiplications in the algorithm results from computations on A, and the other from computations on B, which is key to the proof of the theorem.

THEOREM 4.1. *If A and B result of independent computations, $K(A, B)$ has a martingale-inducing computation DAG.*

Proof. By induction on n.

For $n = 0$ then $K(A, B) = a_0.b_0$. The computation DAG is a single multiplication node and a_0 and b_0 are independent.

For $n \geq 1$, we consider $K(A, B)$ where A and B of degree $2^n - 1$. First, we will show that the partial products P_0, P_1 and P_2 are computed with martingale-inducing DAGs.

Because A and B are independent, so are A_l and B_l. Therefore, $P_0 = K(A_l, B_l)$, where A_l and B_l are of degree $2^{n-1} - 1$ has by induction a martingale-inducing DAG. By the same reasoning, we show that $P_2 = K(A_h, B_h)$ has a martingale-inducing DAG.

For $P_1 = K(A_h - A_l, B_l - B_h)$, A_h and A_l are not necessarily independent, but they are combined using a subtraction operation. Same for B_h and B_l. Moreover, the resulting polynomials $A_h - A_l$ and $B_l - B_h$ are independent of degree $2^{n-1} - 1$. Therefore, by the induction hypothesis, P_1 is computed with a martingale-inducing DAG.

[1] We target polynomials with a number of coefficients that is a power-of-2.

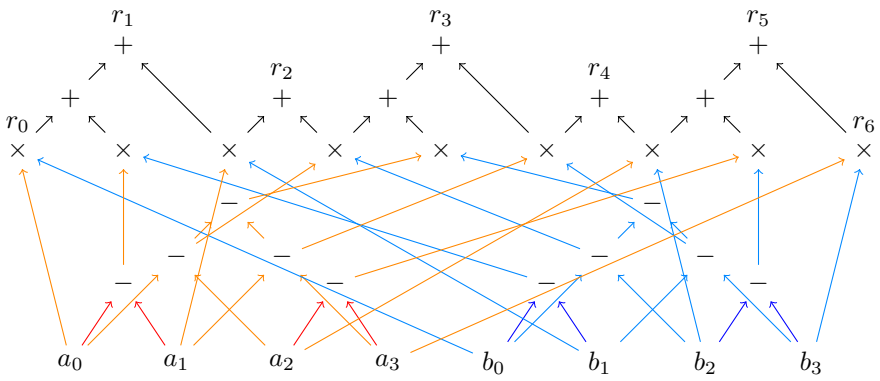


Fig. 3: Computation DAG for the Karatsuba multiplication $R = A \times B$ of two polynomials of degree three — first three levels are the subtractions in the recursive calls, each concerning only one of A or B; then all products are performed, with one operand coming from A (blue) and the other from B (red); finally, zero to two levels of additions yield the result — different hues of blue and red for legibility purpose only.

Finally, $K(A,B) = P_2.X^{2^n} + (P_0 + P_1 + P_2).X^{2^{n-1}} + P_0$. We can ignore the multiplications by X^{2^n} and $X^{2^{n-1}}$, which only shift the position of the coefficients and do not introduce numerical errors.

The coefficients of $K(A,B)$ result of sums of coefficients in P_0, P_1, and P_2. The operands are not always independent because some coefficients are shared, for instance, between P_0 and $P_0.X^{2^{n-1}}$. Nevertheless, all the operations in the resulting DAG are sums. Therefore, we conclude that the computation of $K(A,B)$ is martingale-inducing. ∎

Length of the error martingale. Let us now compute the length of the martingale for each coefficient of the Karatsuba product. For A, B of degree $2^n - 1$, let $R = K(A,B)$ with degree $d = 2^{n+1} - 2$.

$$R = r_{2^{n+1}-2}.X^{2^{n+1}-2} + \ldots + r_1.X + r_0$$

The coefficients of A and B can either be constant inputs or result of previous martingale-inducing computations. We note m_A (respectively m_B) the maximum martingale length of the coefficients of A (respectively B). When coefficients are constant, $m_A = m_B = 0$.

THEOREM 4.2. *For $i \in [\![0, d]\!]$, the length of the error martingale in the computation of coefficient r_i is*

$$m(i,d) = 1 + 3\lfloor \log_2 \min\{i+1, d-i+1\} \rfloor + m_A + m_B$$

Before proving this theorem, let us first give some properties of function $m(i,d)$. Table 1 shows the values of $m(i,d)$ for $n \leq 3$. We note that:

- *(Property 1)* $m(i,d)$ is symmetric with respect to $d/2$, $m(i,d) = m(d-i,d)$.
- *(Property 2)* $m(i,d)$ reaches its maximum for $i = d/2$, $m(d/2,d) = 1 + 3n + m_A + m_B$.

n	d	$m(d,d)\dots m(1,d)\ m(0,d)$
0	0	1
1	2	1 4 1
2	6	1 4 4 7 4 4 1
3	14	1 4 4 7 7 7 7 10 7 7 7 7 4 4 1

Table 1: Values of $m(i,d)$ for $n \leq 3$ and $m_A = m_B = 0$.

- *(Property 3)* The 2^{n-1} coefficients to the left and to the right of the central coefficient $d/2$ have the second largest martingale length:
  $\forall i \in]\!]2^{n-1}-2, d/2[\![\cup]\!]d/2, d-(2^{n-1}-2)[\![,\ m(i,d) = 1+3(n-1)+m_A+m_B$.

It is natural that the error martingale length is smallest for the extreme degrees r_0 and r_d since they result from a single product. On the contrary, the coefficients around $d/2$ are the most sensitive to errors because they result from the sum of many different partial products.

Let us now prove the theorem.

Proof. By induction on n.

For $n = 0$, $d = 0$ and $R = r_0 = a_0.b_0$. Because the DAG is composed of a single multiplication, the length of the error martingale is $1 + m_A + m_B$. We verify that this is the value of $m(0,0)$.

For $n \geq 1$, $d = 2^{n+1} - 2$, we consider R where A and B are of degree $2^n - 1$. Let us first compute the martingale lengths for the partial products P_0, P_1 and P_2:

- $P_0 = K(A_l, B_l)$, where A_l and B_l are of degree $2^{n-1} - 1$. By induction hypothesis, the length of the error martingale is $m_{P_0}(i) = m(i, 2^n - 2)$.
- $P_2 = K(A_h, B_h)$, similarly the length of the error martingale is $m_{P_2}(i) = m(i, 2^n - 2)$.
- $P_1 = K(A_h - A_l, B_l - B_h)$. Each coefficient in $A_h - A_l$ is computed by subtracting two coefficients from A_h and A_l, therefore its maximum martingale length is $1+\max\{m_A, m_A\} = 1+m_A$. The same reasoning applies to $B_h - B_l$, which has a maximum martingale length $1 + m_B$. By induction hypothesis, the length of the error martingale of P_1 is $m_{P_1}(i) = m(i, 2^n - 2) + 2$. As shown, the additional 2 comes from the inner subtractions.

Let us now consider $R = P_2.X^{2^n} + (P_0 + P_1 + P_2).X^{2^{n-1}} + P_0$. Figure 4 represents the shifted partial products in the computation of R : P_0 is not shifted, P_2 is shifted 2^n positions left. $P1$, P_2 and P_0 are shifted 2^{n-1} positions left.

Let us treat separately cases (a), (b), and (c).

Case (a):. For $i \in [\![0, 2^{n-1} - 1]\!]$, r_i corresponds to the i-th coefficient in P_0. Therefore the martingale length for r_i is given by $m_{P_0}(i)$ and

$$m_{P_0}(i) = m(i, 2^n - 2) = 1 + 3\lfloor \log_2(i+1) \rfloor + m_A + m_B = m(i,d)$$

Case (c):. For $i \in [\![d - 2^{n-1} + 1, d]\!]$, r_i corresponds to the $i - 2^n$ coefficient in P_2. Therefore the martingale length for r_i is given by $m_{P_2}(i - 2^n)$ and

$$\begin{aligned} m_{P_2}(i-2^n) = m(i-2^n, 2^n-2) &= 1 + 3\lfloor \log_2\left((2^n-2) - (i-2^n) + 1\right) \rfloor + m_A + m_B \\ &= 1 + 3\lfloor \log_2\left((2^{n+1}-2) - i + 1\right) \rfloor + m_A + m_B \\ &= m(i,d) \end{aligned}$$

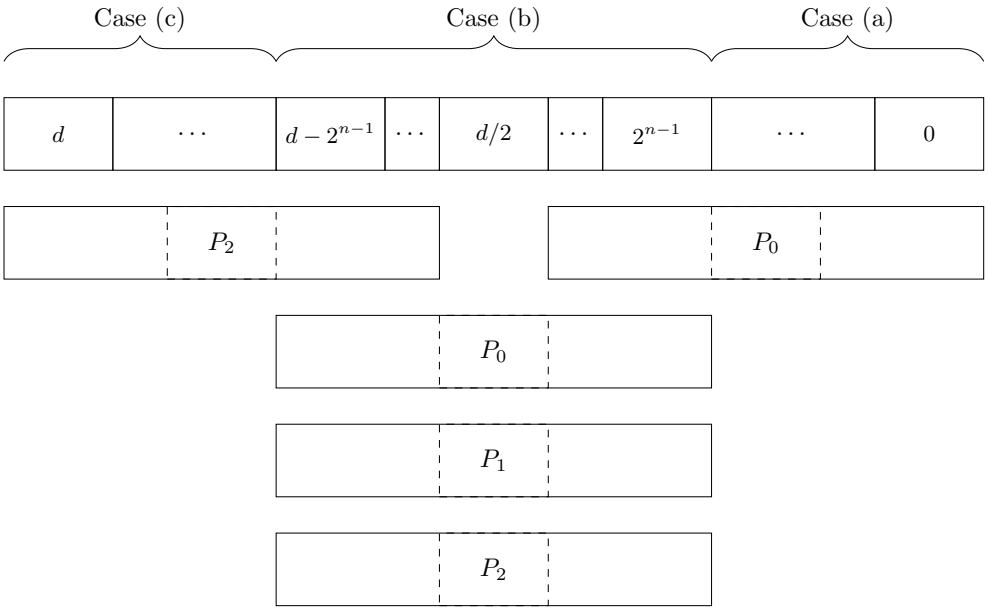


Fig. 4: Shifted partial products in $R = K(A, B)$ with $d = 2^{n+1} - 2$. The dashed cells represent the central coefficient in each polynomial.

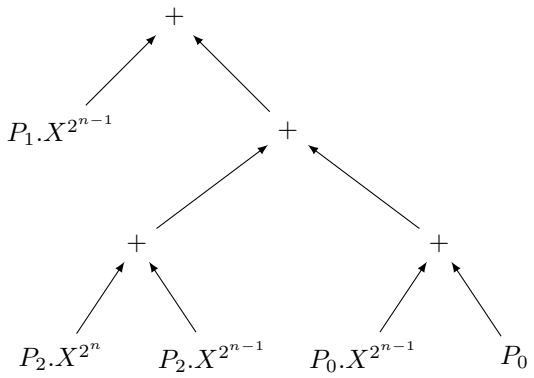


Fig. 5: Summing DAG for case (b) .

Case (b):. This case is the most interesting one, because each coefficient r_i results of the sum of at most four coefficients from P_0, P_2, $P_0.X^{2^{n-1}}$, $P_1.X^{2^{n-1}}$, and $P_2.X^{2^{n-1}}$.

To minimize the martingale length, we will use the summing order from the DAG in Figure 5. Note that even if the figure depicts a tree, it corresponds to a DAG since some leaves share coefficients (e.g. P_2 and $P_2.X^{2^{n-1}}$). Let us note $m_b(i)$ the martingale length of the DAG.

For each node $z \leftarrow x + y$ in the tree, $m_z = 1 + \max\{m_x, m_y\}$. Therefore, the

martingale length for r_i is given by

$$\begin{aligned}
m_b(i) &= 1 + \max\Big\{m_{P_1}(i-2^{n-1}), 1+\max\{ \\
&\qquad 1+\max\{m_{P_2}(i-2^n), m_{P_2}(i-2^{n-1})\}, \\
&\qquad 1+\max\{m_{P_0}(i-2^{n-1}), m_{P_0}(i)\}\}\Big\} \\
&= 1 + \max\Big\{2+m(i-2^{n-1},2^n-2), 1+\max\{ \\
&\qquad 1+\max\{m(i-2^n,2^n-2), m(i-2^{n-1},2^n-2)\}, \\
&\qquad 1+\max\{m(i-2^{n-1},2^n-2), m(i,2^n-2)\}\}\Big\} \\
&= 1 + \max\Big\{2+m(i-2^{n-1},2^n-2), \\
&\qquad 2+\max\{m(i-2^n,2^n-2), m(i-2^{n-1},2^n-2), m(i,2^n-2)\}\Big\} \\
&= 3 + \max\Big\{m(i-2^n,2^n-2), m(i-2^{n-1},2^n-2), m(i,2^n-2)\Big\}
\end{aligned}$$

We note that to achieve a minimal martingale length, it is important to have P_1, which has an additional martingale length of 2, as a direct child of the root node.

Now let us compute $m_b(i)$:

- For $i = d/2 = 2^n - 1$, the maximum is reached for P_0, P_1, and P_2 due to *Property 2*,

$$\begin{aligned}
m_b(i) &= 3 + m(2^n - 1 - 2^{n-1}, 2^n - 2) = 3 + m(2^{n-1} - 1, 2^n - 2) \\
&= 3 + 3(n-1) + 1 + m_A + m_B = 1 + 3n + m_A + m_B = m(i,d)
\end{aligned}$$

- For $i \in [\![2^{n-1}, d/2[\![$, we subdivide the interval in two.
  - For $i \in [\![2^{n-1}, 2^{n-1} + 2^{n-2}[\![$, we are close to the central element of P_0, therefore due to *Property 3*

$$\begin{aligned}
m_b(i) &= 3 + 3(n-2) + 1 + m_A + m_B = 3(n-1) + 1 + m_A + m_b \\
&= m(i,d)
\end{aligned}$$

  - For $i \in [\![2^{n-1} + 2^{n-2}, d/2[\![$, we are close to the central element of P_1 (or $-P_2, -P_0$), therefore due to *Property 3* we conclude as before.
- For $i \in]\!]d/2, d - 2^{n-1}]\!]$, we apply a similar proof scheme by subdividing the interval and applying *Property 3*. ∎

Finally, for all $i \in [\![0, d]\!]$, Corollaries 3.3 and 3.6 show that

$$\frac{|\hat{r_i} - r_i|}{|r_i|} \leq \mathcal{K}\sqrt{u\gamma_{2(3\lfloor \log_2(i+1)\rfloor)}(u)}\sqrt{\ln(2/\lambda)} \tag{4.3}$$

with probability at least $1 - \lambda$. The bound is maximal for the central coefficient $i = d/2$ (due to *Property 1*) with $d = 2^{n+1} - 2$,

$$\frac{\left|\hat{r}_{d/2} - r_{d/2}\right|}{\left|r_{d/2}\right|} \leq \mathcal{K}\sqrt{u\gamma_{6n}(u)}\sqrt{\ln(2/\lambda)}. \tag{4.4}$$

We perform numerical experiments for d varying from 3 to $2^{16} - 1$. The computation is performed in IEEE-754 RN-binary32 and SR-nearness-binary32. Errors

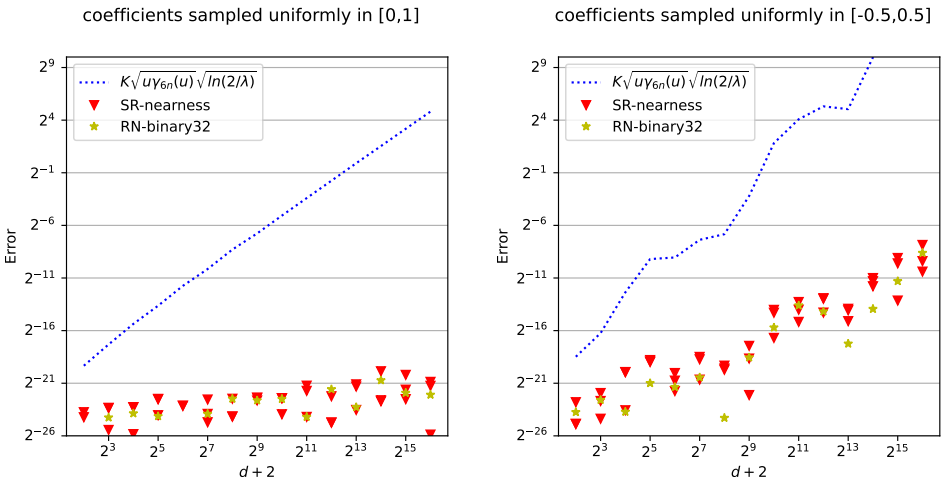


Fig. 6: Relative error for the subtractive Karatsuba algorithm. In the left plot, coefficients were uniformly sampled in $[0, 1]$. In the right plot, coefficients were uniformly sampled in $[-0.5, 0.5]$. ($1 - \lambda = 0.9$ and $d = 2^{n+1} - 2$).

are computed for the central coefficient $d/2$ against a IEEE-754 binary64 reference. For each degree, three SR samples are computed with Verificarlo [4]. The condition number bound, K, is computed following the lemmas 3.2 and 3.5.

First, we consider polynomials with positive coefficients uniformly sampled in $[0, 1]$ in Figure 6. The bound growth is dominated by K, which grows linearly with d. Despite this, the actual error grows slowly for these inputs and stays under 2^{-20}.

Then, we consider polynomials with coefficients uniformly sampled in $[-0.5, 0.5]$. The condition number still dominates the bound. When we have both positive and negative coefficients, catastrophic cancellations between terms trigger often, accounting for the faster growth of K. We observe that the SR and RN samples also show this effect, with a higher error than before: for $d = 2^{16} - 1$, the relative error is around 2^{-11}.

Unlike the previous examples in this section, for which the condition number was 1 for positive inputs, Karatsuba multiplication has a K that grows linearly with the input size which explains the loose bound. This growth happens because of the crossed-product terms in P_1 that are later cancelled in the final sum with P_0, P_1, P_2.

Karatsuba multiplication experiments indicate that the condition number can significantly influence the error and the bound. Current approaches tend to focus primarily on minimizing the error while relying on worst-case bounds for inputs. This observation prompts further investigations into the rounding error analysis of algorithms with SR.

5. Doob-Meyer decomposition and non-linear errors. To establish a comprehensive framework, we propose using the Doob-Meyer decomposition, a central result in the study of stochastic processes [5, p 296]. This decomposition separates a stochastic process into two distinct components: a martingale part and a predictable process. Let us first recall the definition of a predictable stochastic process [3, p 65].

DEFINITION 2. *Given a filtration $(\mathbb{F}_n)_{n\geq 0}$, a stochastic process X_n is predictable*

if X_0 is $\mathbb{F}_0$-measurable, and X_n is $\mathbb{F}_{n-1}$-measurable for all $n \geq 1$.

This means that the value of X_n is known at the previous time step. Now, let us state the Doob–Meyer decomposition.

THEOREM 5.1 (Doob–Meyer decomposition). *Let $(\mathbb{F}_k)_{0\leq k\leq n}$ and $X_0, \ldots, X_n$ an adapted stochastic process locally integrable, meaning that $E(|X_k|) < \infty$ for all $0 \leq k \leq n$. There exists a martingale $M_0, \ldots, M_n$ and a predictable integrable sequence $A_0, \ldots, A_n$ starting with $A_0 = 0$ for which we have:*

$$\begin{cases} X_n & = M_n + A_n, \\ A_n & = \sum_{k=1}^{n} E[X_k - X_{k-1}/\mathbb{F}_{k-1}], \\ E(M_n) & = 0. \end{cases}$$

This decomposition is almost surely unique.

The martingale M_n reflects the information available up to time n. It does not exhibit any drift and captures the unbiased random component of the stochastic process X_n. While the sequence A_n represents the cumulative effect of the predictable part of the stochastic process X_n. It can be interpreted as the drift of X_n. Its predictability means that, at each step, one knows the value of the drift at the next step. For instance, at a step when the algorithm squares a value with error, a term square of the current error will be added to the drift, while the martingale remains centered on 0.

We propose to use Doob-Meyer decomposition to analyze the error under SR-nearness. We consider an algorithm executed under SR-nearness. Its error is a stochastic process $X = \hat{y} - y$. Because each random error δ_i is bounded $|\delta_i| \leq u$, the resulting stochastic process X must also be bounded and is locally integrable. Therefore we can apply Doob-Meyer decomposition and write the error as the sum of a martingale and a drift:

$$\hat{y} - y = M + A, \tag{5.1}$$

The martingale component in Equation (5.1) captures the unbiased stochastic behavior; in other words, the errors that can be compensated with SR, while the bias is the expected last term of the drift.

Multi-linear error, $A = 0$. In this paper, we study algorithms whose computation graphs are martingale-inducing DAGs. For these algorithms, the error terms are always of degree one, which is why we described them as having multi-linear errors. In fact, in a martingale-inducing DAG, the product of two nodes is allowed only if they have different errors, preventing any increase in the degree of the errors. Furthermore, the addition operation does not increase the degree of errors, even if the two operands share some errors.

In the case of multi-linear errors algorithms, as shown in Section 3, the forward error is always captured by a martingale, therefore for multi-linear error the drift component A is zero.

Non-linear error. For non-linear error algorithms, we cannot apply the method from Section 3. Nevertheless, Doob-Meyer decomposition still applies and provides a simplifying framework for analyzing the error. Indeed, the martingale term can be studied with Azuma-Hoeffding and has a probabilistic bound in $\mathcal{O}(\sqrt{n}u)$. The problem is, therefore, reduced to the study of the drift term A.

In [10], we have studied variance computation algorithms. By deterministically bounding the drift term A, we showed that it was negligible at the first order over

u and proved an error bound in $\mathcal{O}(\sqrt{n}u)$. El Arar et al. [7, thm 3] have implicitly used this decomposition to study the effect of the number of random bits required to implement SR effectively. We conjecture that the drift is negligible when $nu^2 = o(1)$. However, for low precision computations, the drift may have a dominant effect on the precision of the result.

Doob-Meyer provides an interesting decomposition for analyzing non-linear algorithms. Nevertheless, in general, it is not easy to build the decomposition, and bounding the error of general non-linear algorithms under SR-nearness remains an open problem.

6. Conclusion. The worst-case error bound for a computation involving n elementary operations is $\mathcal{O}(nu)$. This bound, while useful, can be overly pessimistic as it assumes a deterministic accumulation of errors and does not account for error compensation phenomena. With SR, the use of probabilistic tools, including martingales, variance analysis, and concentration inequalities, allows us to better investigate rounding errors behavior, establish probabilistic error bounds in $\mathcal{O}(\sqrt{n}u)$.

In this paper, we propose a general methodology to build a martingale for any computation DAG with multi-linear errors arising from addition, subtraction, and multiplication operations. We applied this methodology to pairwise summation and Horner algorithms, confirming results consistent with earlier works on these algorithms under SR. Moreover, to the best of our knowledge, we are the first to analyze Karatsuba polynomial multiplication under SR. Using our approach, we established a probabilistic error bound in $\mathcal{O}(\sqrt{n}u)$ for Karatsuba's algorithm as well. We have also discussed how to analyze the error of a general algorithm using the Doob-Meyer decomposition that separates the martingale term and the drift part. We believe that this probabilistic framework can serve as an effective tool to improve the rounding error analysis under SR in numerical algorithms.

The scripts to reproduce the numerical experiments of Section 4.3 are made available at https://github.com/verificarlo/sr-karatsuba.

REFERENCES

[1] M. P. Connolly, N. J. Higham, and T. Mary, *Stochastic rounding and its probabilistic backward error analysis*, SIAM Journal on Scientific Computing, (2021).

[2] M. Croci, M. Fasi, N. J. Higham, T. Mary, and M. Mikaitis, *Stochastic rounding: implementation, error analysis and applications*, Royal Society Open Science, 9 (2022), p. 211631.

[3] D. Dacunha-Castelle, D. McHale, and M. Duflo, *Probability and Statistics: Volume II*, no. v. 2, Springer New York, 2012.

[4] C. Denis, P. de Oliveira Castro, and E. Petit, *Verificarlo: Checking floating point accuracy through Monte Carlo arithmetic*, in 23nd IEEE Symposium on Computer Arithmetic, ARITH 2016, Silicon Valley, CA, USA, July 10-13, 2016, 2016, pp. 55–62.

[5] J. Doob, *Stochastic Processes*, Probability and Statistics Series, Wiley, 1953.

[6] E.-M. El Arar, *Stochastic models for the evaluation of numerical errors*, PhD thesis, Université Paris-Saclay, 2023.

[7] E.-M. El Arar, M. Fasi, S.-I. Filip, and M. Mikaitis, *Probabilistic error analysis of limited-precision stochastic rounding*, 2024, https://arxiv.org/abs/2408.03069.

[8] E.-M. El Arar, D. Sohier, P. de Oliveira Castro, and E. Petit, *The positive effects of stochastic rounding in numerical algorithms*, in 2022 IEEE 29th Symposium on Computer Arithmetic (ARITH), 2022, pp. 58–65.

[9] E.-M. El Arar, D. Sohier, P. de Oliveira Castro, and E. Petit, *Stochastic rounding variance and probabilistic bounds: A new approach*, SIAM Journal on Scientific Computing, 45 (2023), pp. C255–C275.

[10] E.-M. El Arar, D. Sohier, P. de Oliveira Castro, and E. Petit, *Bounds on nonlinear errors for variance computation with stochastic rounding*, SIAM Journal on Scientific

Computing, 46 (2024), pp. B579–B599.
[11] S. Gupta, A. Agrawal, K. Gopalakrishnan, and P. Narayanan, *Deep learning with limited numerical precision*, in International conference on machine learning, PMLR, 2015, pp. 1737–1746.
[12] E. Hallman and I. C. F. Ipsen, *Precision-aware deterministic and probabilistic error bounds for floating point summation*, Numerische Mathematik, 155 (2023), pp. 83–119.
[13] I. C. F. Ipsen and H. Zhou, *Probabilistic error analysis for inner products*, SIAM Journal on Matrix Analysis and Applications, 41 (2020), pp. 1726–1741.
[14] A. A. Karatsuba, *The complexity of computations*, Proceedings of the Steklov Institute of Mathematics-Interperiodica Translation, 211 (1995), pp. 169–183.
[15] M. Mitzenmacher and E. Upfal, *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*, Cambridge University Press, 2005.
[16] J.-M. Muller, N. Brisebarre, F. De Dinechin, C.-P. Jeannerod, V. Lefevre, G. Melquiond, N. Revol, D. Stehlé, S. Torres, et al., *Handbook of floating-point arithmetic*, vol. 1, Birkhäuser Basel, 2nd ed., 2018.
[17] L. Xia, M. E. Hochstenbach, and S. Massei, *On the convergence of the gradient descent method with stochastic fixed-point rounding errors under the polyak-lojasiewicz inequality*, arXiv preprint arXiv:2301.09511, (2023).
[18] L. Xia, S. Massei, M. E. Hochstenbach, and B. Koren, *On the influence of stochastic roundoff errors and their bias on the convergence of the gradient descent method with low-precision floating-point computation*, 2023, https://arxiv.org/abs/2202.12276.