Variance Arithmetic*

Chengpu Wang
40 Grossman Street, Melville, NY 11747, USA
Chengpu@gmail.com

2024/04/17

Abstract

A new deterministic uncertainty-bearing floating-point arithmetic called *variance arithmetic* is developed to predict the result uncertainty variance for analytic functions. It is based on the central limit theorem and independence of input uncertainties. It uses Taylor expansion to predict the result value bias and uncertainty variance due to input uncertainty variances, and such prediction is precise if the input uncertainty variances are all precise. For floating-point rounding errors, it predicts the result uncertainty variance numerically within approximately 4-fold of the actual result uncertainty variance.

Variance arithmetic tracks the correlations of input within analytic expression using standard statistics through Taylor expansion. Due to this dependency tracing, variance arithmetic allows no numerical execution freedom, and it has no dependency on such freedoms. Variance arithmetic rejects certain calculations such as inverting an input near zero on the mathematical ground of the divergence of the result uncertainty variance, which is also the statistical ground that either a distributional pole or a distributional zero is in the input bounding ranges statistically.

Variance arithmetic has been showcased to be widely applicable, such as for analytic calculations, progressive calculation, regressive generalization, polynomial expansion, statistical sampling, and transformations.

Keywords: computer arithmetic, error analysis, interval arithmetic, uncertainty, numerical algorithms.
AMS subject classifications: G.1.0

Contents

*

1 Introduction

1.1 Measurement Uncertainty

Except for the simplest counting, scientific and engineering measurements never give completely precise results [2][3]. In scientific and engineering measurements, the uncertainty of a measurement x usually is characterized by either the sample deviation δx or the uncertainty range Δx [2][3].

- If $\delta x = 0$ or $\Delta x = 0$, x is a *precise value*.
- Otherwise, x is an *imprecise value*.

$P \equiv \delta x/|x|$ is defined as the *statistical precision* (or simply precision in this paper) of the measurement, in which x is the value, and δx is the uncertainty deviation. A larger precision means a coarser measurement while a smaller precision mean finer measurement. The precision of measured values ranges from an order-of-magnitude estimation of astronomical measurements to 10^{-2} to 10^{-4} of common measurements to 10^{-14} of state-of-art measurements of basic physics constants [4].

1.2 Conventional Floating-Point Arithmetic

The *conventional floating-point arithmetic* [5][6][7] assumes a constant and best possible precision for each value all the time, and constantly generates artificial information during the calculation [8]. For example, the following calculation is carried out precisely in integer format:

$$64919121 \times 205117922 - 159018721 \times 83739041 = \\ 13316075197586562 - 13316075197586561 = 1; \quad (1.1)$$

If Formula (1.1) is carried out using conventional floating-point arithmetic:

$$64919121 \times 205117922 - 159018721 \times 83739041 = \\ 64919121.000000000 \times 205117922.000000000 - 159018721.000000000 \times 83739041.000000000 = \\ 13316075197586562. - 13316075197586560. = 2. = 2.0000000000000000; \quad (1.2)$$

1. The multiplication results exceed the maximal significance of the 64-bit IEEE floating-point representation. They are rounded off, generating rounding errors;
2. The normalization of the subtraction result amplifies the rounding error to most significant bit (MSB) by padding zeros, to result in catastrophic cancellation [9] [10] of this case.

Because a rounding error from lower digits quickly propagates to higher digits, the 10^{-7} precision of the 32-bit IEEE floating-point format [5][6][7] is usually not fine enough for calculations involving input data of 10^{-2} to 10^{-4} precision.

Self-censored rules are developed to avoid such rounding error propagation [11][12], such as avoiding subtracting results of large multiplication, as in Formula (1.2). However, these rules are not enforceable, and in many cases are difficult to follow, and even more difficult to quantify. So far, the study of rounding error propagation is concentrated on linear calculations [9] [10] [13], or on special cases [11] [14] [15], while the rounding errors are generally manifested as the lack of numerical stability [16]. For the existed systematic studies of rounding errors:

- The forward rounding error [10] compares the result with rounding error and the ideal result without rounding error, such as comparing the result using 32-bit IEEE floating-point arithmetic and the corresponding result using 32-bit IEEE floating-point arithmetic [17]. The most recent such study gives a very optimistic view on numerical library errors as a fraction of the least significant bit of the significand of floating-point results [17]. However, it is questionable that the result using 64-bit IEEE floating-point arithmetic is more accurate than the corresponding result using 32-bit IEEE floating-point arithmetic. e.g., for Taylor expansion of $\sin(x)$, using 64-bit IEEE floating-point arithmetic results in more expansion terms so that possibly more least significant bits which are affected by rounding errors. Such study usually does not evaluate the numerical library errors in other way, such as comparing $\tan(x)$ and $\sin(x)/\cos(x)$. This paper will show that the numerical errors for $\sin(x)$, $cos(x)$, and $\tan(x)$ are much worse than the least significant bit of the significand of floating-point results.
- The backward rounding error [9] [10] [13] only estimates the result uncertainty due to rounding errors, so that it misses the bias caused by rounding errors. Such analysis is generally limited to very small uncertainty due to its usage of perturbation theory, and is specific for for each specific algorithm [9] [10] [13]. In contrast, variance arithmetic is generic for any analytic function, and it can deal with uncertainties of any magnitudes. Variance arithmetic also provides generic calculation of uncertainty bias, which is significant for calculations in general.

The conventional numerical approaches explore the path-dependency property of calculations based on conventional floating-point arithmetic, to find the best possible methods [10] [11] [12]. However, variance arithmetic challenges this conventional wisdom by demonstrating the path independence requirement for result uncertainties.

1.3 Interval Arithmetic

Interval arithmetic [12][18][19][20][21][22] is currently a standard method to track calculation uncertainty. It ensures that the value x is absolutely bounded within its *bounding range* $[x] \equiv [\underline{\mathrm{x}}, \bar{x}]$, in which $\underline{\mathrm{x}}$ and $\bar{x}$ are lower and upper bounds for x, respectively. In this paper, interval arithmetic is simplified and tested as the following arithmetic formulas[1] [20]:

$$[x] + [y] = [\underline{\mathrm{x}} + \underline{\mathrm{y}}, \bar{x} + \bar{y}]\,; \tag{1.3}$$

$$[x] - [y] = [\underline{\mathrm{x}} - \bar{y}, \bar{x} - \underline{\mathrm{y}}]\,; \tag{1.4}$$

$$[x] \times [y] = [\min(\underline{\mathrm{x}}\underline{\mathrm{y}}, \underline{\mathrm{x}}\bar{y}, \bar{x}\underline{\mathrm{y}}, \bar{x}\bar{y}), \max(\underline{\mathrm{x}}\underline{\mathrm{y}}, \underline{\mathrm{x}}\bar{y}, \bar{x}\underline{\mathrm{y}}, \bar{x}\bar{y})]\,; \tag{1.5}$$

$$0 \notin [y] :\ [x]\,/\,[y] = [x] \times [1/\bar{y}, 1/\underline{\mathrm{y}}]\,; \tag{1.6}$$

If interval arithmetic is implemented using a floating-point representation with limited resolution, its resulting bounding range is widened further [19].

A basic problem is that the bounding range used by interval arithmetic is not compatible with usual scientific and engineering measurements, which instead use the statistical mean and deviation to characterize uncertainty [2][3]. Most measured values are well approximated by a Gaussian distribution [2][3][23], which has no limited bounding range. Let *bounding leakage* be defined as the possibility of the true value

[1] For the mathematical definition of interval arithmetic, please see [22].

to be outside a bounding range. If a bounding range is defined using a statistical rule on bounding leakage, such as the $6\sigma - 10^{-9}$ rule for Gaussian distribution [23] (which says that the bounding leakage is about 10^{-9} for a bounding range of mean $\pm$ 6-fold of standard deviations), there is no guarantee that the calculation result will also obey the $6\sigma - 10^{-9}$ rule using interval arithmetic, since interval arithmetic has no statistical foundation [2]

Another problem is that interval arithmetic only provides the worst case of uncertainty propagation, so that it tends to over-estimate uncertainty. For instance, in addition and subtraction, it gives the result when the two operands are +1 and -1 correlated respectively [25]. However, if the two operands are -1 and +1 correlated respectively instead, the actual bounding range after addition and subtraction reduces, which is called the best case in random interval arithmetic [26]. The vast overestimation of bounding ranges in these two worst cases prompts the development of affine arithmetic [25][27], which traces error sources using a first-order model. Being expensive in execution and depending on approximate modeling even for such basic operations as multiplication and division, affine arithmetic has not been widely used. In another approach, random interval arithmetic [26] reduces the uncertainty over-estimation of standard interval arithmetic by randomly choosing between the best-case and the worst-case intervals.

A third problem is that the results of interval arithmetic may depend strongly on the actual expression of an analytic function $f(x)$. For example, Formula (1.7), Formula (1.8) and Formula (1.9) are different expressions of the same $f(x)$; however, the correct result is obtained only through Formula (1.7), and uncertainty may be exaggerated in the other two forms, e.g., by 67-fold and 33-fold at input range [0.49, 0.51] using Formula (1.8) and Formula (1.9), respectively. This is called the dependence problem of interval arithmetic [21]. There is no known generic way to apply Taylor expansion in interval arithmetic, so that the dependence problem is an intrinsic problem for interval arithmetic.

$$f(x) = (x - 1/2)^2 - 1/4; \tag{1.7}$$

$$f(x) = x^2 - x; \tag{1.8}$$

$$f(x) = (x - 1)x; \tag{1.9}$$

A forth problem is that interval arithmetic lacks mechanism to reject an calculation, while a mathematical calculation always has a valid input range for a given targeted domain. For example, when an input interval contains zero:

- For square root, interval arithmetic divides the interval into two, to provide a real answer and an imaginary answer. Such branching is uncommon because every application has its limited domain.
- For inversion, interval arithmetic divides the interval into two, to provide two answers both contain infinities. While traditionally the infinitive result is usually considered as invalid.

[2] There is some attempt [24] to connect intervals in interval arithmetic to confidence interval or the equivalent so-called p-box in statistics. Because this attempt seems to rely heavily on 1) specific properties of the uncertainty distribution within the interval and/or 2) specific properties of the functions upon which the interval arithmetic is used, this attempt does not seem to be generic. If probability model is introduced to interval arithmetic to allow tiny bounding leakage, the bounding range is much less than the corresponding pure bounding range [13]. Anyway, these attempts seem to be outside the main course of interval arithmetic, which has no statistics in mind.

It is quite possible that an interval containing zero is unsuitable for either operations. Although new rules can be added to interval arithmetic to reject certain results, such addition is not natural, unlike the mathematical criteria to judge the divergence of a calculations.

Interval arithmetic has very coarse and algorithm-specific precision but constant zero bounding leakage. It represents the other extreme from conventional floating-point arithmetic. To meet practical needs, a better uncertainty-bearing arithmetic should be based on statistical propagation of the rounding error, while also allowing reasonable bounding leakage for normal usages, which is achieved in variance arithmetic, as show later in this paper.

As shown previously [1], interval arithmetic grossly exaggerates calculation uncertainty, e.g., for $f^{-1}(f(x))$ the result uncertainty is much larger than the original uncertainty. On the other hand, as shown later in this paper, variance arithmetic provides stable bounding for a given bounding leakage, e.g., ± 5-fold of the result deviation from the mean for a bounding leakage of 5.73×10^{-7}. Variance arithmetic is based on statistics. It has no dependency problem. Its statistical context rejects certain input intervals, such as inversion and square root of an interval containing zero.

1.4 Statistical Propagation of Uncertainty

If each operand is regarded as a random variable, and the statistical correlation between the two operands is known, the resulting uncertainty is given by the *statistical propagation of uncertainty* [28][29], with the following arithmetic equations, in which σ is the deviation of a measured value x, P is its precision, and γ is the correlation between the two imprecise values:

$$(x \pm \delta x) + (y \pm \delta y) = (x + y) \qquad \pm\sqrt{\delta x^2 + \delta y^2 + 2\delta x \delta y \gamma}; \tag{1.10}$$

$$(x \pm \delta x) - (y \pm \delta y) = (x - y) \qquad \pm\sqrt{\delta x^2 + \delta y^2 - 2\delta x \delta y \gamma}; \tag{1.11}$$

$$(x \pm \delta x) \times (y \pm \delta y) = (x \times y) \qquad \pm|x \times y|\sqrt{{P_x}^2 + P(y)^2 + 2P_x P(y)\gamma}; \tag{1.12}$$

$$(x \pm \delta x)/(y \pm \delta y) = (x/y) \qquad \pm|x/y|\sqrt{{P_x}^2 + P(y)^2 - 2P_x P(y)\gamma}; \tag{1.13}$$

Tracking uncertainty propagation statistically seems a better solution. However, in practice, the correlation between two operands is generally not precisely known, so the direct use of statistical propagation of uncertainty is very limited. *As show later in this paper, variance arithmetic is based on a statistical assumption much more lenient than knowing the correlation between imprecise values.*

1.5 Significance Arithmetic [1]

Significance arithmetic [30] tries to track reliable bits in an imprecise value during the calculation. In the two early attempts [31][32], the implementations of significance arithmetic are based on simple operating rules upon reliable bit counts, rather than on formal statistical approaches. They both treat the reliable bit counts as integers when applying their rules, while in reality, a reliable bit count could be a fractional number [33], so they both can cause artificial quantum reduction of significance. The significance arithmetic marketed by Mathematica [33] uses a linear error model that is consistent with a first-order approximation of interval arithmetic [12][20][21], and

further provides an arbitrary precision representation which is in the framework of conventional floating-point arithmetic. It is definitely not a statistical approach.

Stochastic arithmetic [34] [35], which can also be categorized as significance arithmetic, randomizes the least significant bits (LSB) of each of input floating-point values, repeats the same calculation multiple times, and then uses statistics to seek invariant digits among the calculation results as significant digits. This approach may require too much calculation since the number of necessary repeats for each input is specific to each algorithm, especially when the algorithm contains branches. Its sampling approach may be more time-consuming and less accurate than direct statistical characterization [23], such as directly calculating the mean and deviation of the underlying distribution. It is based on modeling rounding errors in conventional floating-point arithmetic, which is quite complicated. A better approach may be to define arithmetic rules that make error tracking by probability easier.

One problem of significance arithmetic is that itself can not properly specify the uncertainty [1]. For example, if the least significand bit of significand is used to specify uncertainty, then the representation has very coarse precision, such as $1 \pm 10^{-3} = 1024 \times 2^{-10}$ *[1]. Introducing limited bits calculated inside uncertainty cannot avoid this problem completely. Thus, the resolution of the conventional floating-point representation is desired. This is the reason why variance arithmetic has abandoned the significance arithmetic nature of its predecessor [1].*

1.6 An Overview of This Paper

In this paper, a new floating-point arithmetic called *variance arithmetic* is developed to track uncertainty during floating-point calculations of analytic functions:

1. Section 1 compares variance arithmetic with each known uncertainty-bearing arithmetic.
2. Section 2 contains the theoretical foundation for variance arithmetic.
3. Section 3 contains standard and methodology to validate variance arithmetic.
4. Section 4 presents a new criterion to truncate an expansion.
5. Section 5 applies variance arithmetic to adjugate matrix and matrix inversion.
6. Section 6 applies variance arithmetic to process time-series data.
7. Section 7 associates result convergence in variance arithmetic with distributional pole and distributional zero.
8. Section 8 tests variance arithmetic for the most common math library functions.
9. Section 9 focuses on the effect of numerical library errors and shows that such library errors can be very significant, especially when they resonate with input data.
10. Section 10 applies variance arithmetic to regression.
11. Section 11 provides a summary and some discussion for variance arithmetic.

By applying an uncertainty-bearing arithmetic to analytic problem for the first time, this paper breaks ground in the following areas:

- a well-defined mechanism to validate uncertainty-bearing inputs for an analytic function;
- ability to trace dependence using Taylor expansion, which is an expansion of the conventional variance-covariance matrix [23];

- the bias to a result value due to the input uncertainties;
- the invalidation of conventional numerical approaches which will always have the dependency problems;
- a well defined validation method for the uncertainty-bearing result; and
- the need to recalculate numerical libraries with uncertainties as part of the result.

2 Variance arithmetic

2.1 Foundation for Variance Arithmetic

The statistical precision $P(x) = \delta x/|x|$ represents the reliable information content of a measurement statistically, with finer or smaller statistical precision means higher reliability and better reproducibility of the measurement [2][3]. When $1 \leq P(x)$, either δx is too coarse to determine the actual value of x, or $x \pm \delta x$ is a measurement of 0.

The inputs to variance arithmetic should obey the *uncorrelated uncertainty assumption* [1], which means that the uncertainty of any input can be regarded as uncorrelated of any other input uncertainty and all input values . This assumption is consistent with the common methods in processing experimental data [2][3], such as the common knowledge or belief that precision improves with the count n as $1/\sqrt{n}$ during averaging. It is much weaker than requiring any two different inputs to be independent of each other. It can be turned into a realistic and simple statistical requirement or test between two inputs.

Taylor expansion can be used to calculate the result variance of analytic functions.

The uncorrelated uncertainty assumption only applies to imprecise inputs. When the inputs obey the *uncorrelated uncertainty assumption*, in the intermediate steps, variance arithmetic uses statistics to account for the correlation between uncertainties through their associated variables. Therefore, variance arithmetic has no dependency problem.

2.2 The Uncorrelated Uncertainty Assumption [1]

When there is a good estimation of the sources of uncertainty, the uncorrelated uncertainty assumption can be judged directly, e.g., if noise [2][3] is the major source of uncertainty, the uncorrelated uncertainty assumption is probably true. This criterion is necessary to ascertain repeated measurements of the same signal. Otherwise, the uncorrelated uncertainty assumption can be judged by the correlation and the respectively precision of two measurements.

The correlated parts common to different measurements are regarded as signals, which can either be desired or unwanted. Let X, Y, and Z denote three mutually independent random variables [23] with variance $V(X)$, $V(Y)$ and $V(Z)$, respectively. Let α denote a constant. Let $Cov()$ denote the covariance function. Let γ denote the correlation between $(X+Y)$ and $(\alpha X+Z)$. And let:

$$\eta_1^2 \equiv \frac{V(Y)}{V(X)}; \quad \eta_2^2 \equiv \frac{V(Z)}{V(\alpha X)} = \frac{V(Z)}{\alpha^2 V(X)}; \tag{2.1}$$

$$\gamma = \frac{Cov(X+Y, \alpha X+Z)}{\sqrt{V(X+Y)}\sqrt{V(\alpha X+Z)}} = \frac{\alpha/|\alpha|}{\sqrt{1+\eta_1^2}\sqrt{1+\eta_2^2}} \equiv \frac{\alpha/|\alpha|}{1+\eta^2}; \tag{2.2}$$

Formula (2.2) gives the correlation γ between two random variables, each of which contains a completely uncorrelated part and a completely correlated part X, with η being the average ratio between these two parts. Formula (2.2) can also be interpreted reversely: if two random variables are correlated by γ, each of them can be viewed hypothetically as containing a completely uncorrelated part and a completely correlated part, with η being the average ratio between these two parts [3].

[3] The widely used common factor analysis is also based on this principle.

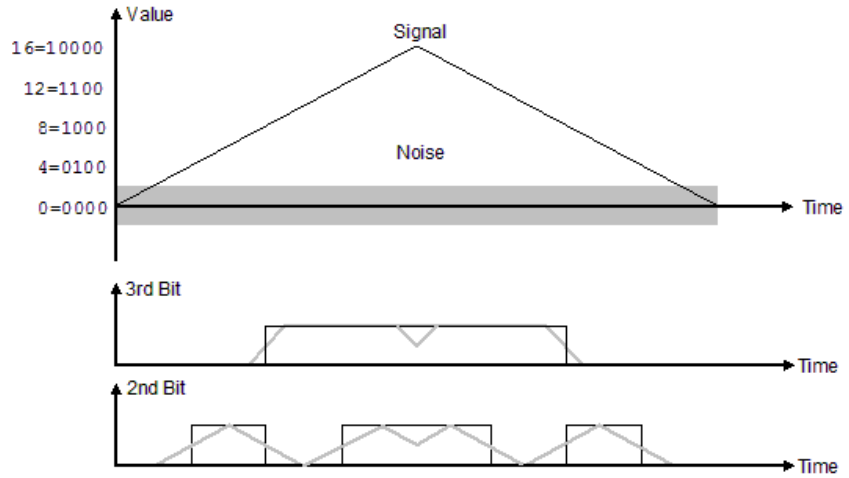

Figure 1: Effect of noise on bit values of a measured value. The triangular wave signal and the added white noise are shown at top using the thin black line and the grey area, respectively. The values are measured by a theoretical 4-bit Digital-to-Analog Converter in ideal condition, assuming LSB is the 0th bit. The measured 3rd and 2nd bits without the added noise are shown using thin black lines, while the mean values of the measured 3rd and 2nd bits with the added noise are shown using thin grey lines. This figure has been published previously [1].

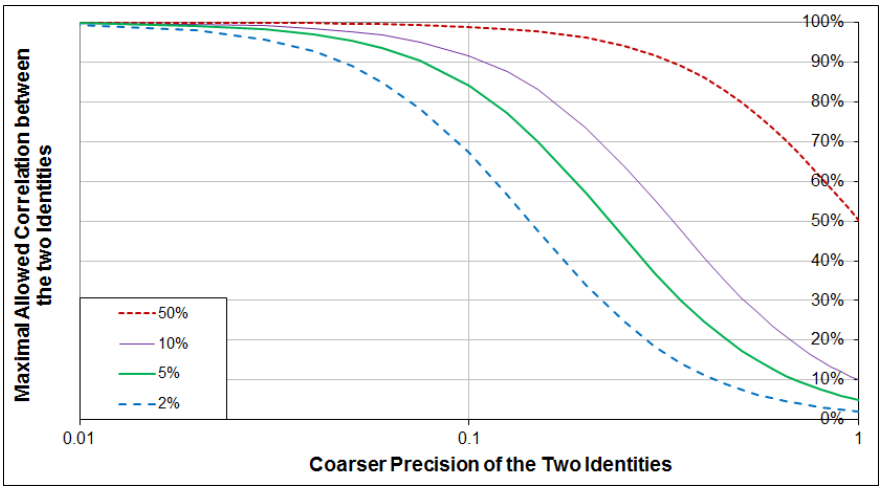

Figure 2: Allowed maximal correlation between two values vs. input precisions and independence standard (as shown in legend) for the independence uncertainty assumption of variance arithmetic to be true. This figure has been published previously [1].

One special application of Formula (2.2) is the correlation between a measured signal and its true signal, in which noise is the uncorrelated part between the two. Figure 1 shows the effect of noise on the most significant two bits of a 4-bit measured signal when $\eta = 1/4$. Its top chart shows a triangular waveform between 0 and 16 as a black line, and a white noise between -2 and +2, using the grey area. The measured signal is the sum of the triangle waveform and the noise. The middle chart of Figure 1 shows the values of the 3rd digit of the true signal as a black line, and the mean values of the 3rd bit of the measurement as a grey line. The 3rd bit is affected by the noise during its transition between 0 and 1. For example, when the signal is slightly below 8, only a small positive noise can turn the 3rd digit from 0 to 1. The bottom chart of Figure 1 shows the values of the 2nd digit of the signal and the measurement as a black line and a grey line, respectively. Figure 1 clearly shows that the correlation between the measurement and the true signal is less at the 2nd digit than at the 3rd digit. Quantitatively, according to Formula (2.2):

1. The overall measurement is 99.2% correlated to the signal with $\eta = 1/8$;
2. The 3rd digit of the measurement is 97.0% correlated to the signal with $\eta = 1/4$;
3. The 2nd digit of the measurement is 89.4% correlated to the signal with $\eta = 1/2$;
4. The 1st digit of the measurement is 70.7% correlated to the signal with $\eta = 1$;
5. The 0th digit of the measurement is 44.7% correlated to the signal with $\eta = 2$.

The above conclusion agrees with the common experiences that, below the noise level of measured signals, noises rather than true signals dominate each digit.

Similarly, while the correlated portion between two values has the same value at each bit of the two values, the ratio of the uncorrelated portion to the correlated portion increases by 2-fold for each bit down from MSB of the two values. Quantitatively, let P denote the precision of an imprecise value, and let η_P denote the ratio of the uncorrelated portion to the correlated portion at level of uncertainty; then η_P increases with decreased P according to Formula (2.3). According to Formula (2.2), if two significant values are overall correlated with γ, at the level of uncertainty the correlation between the two values decreases to γ_P according to Formula (2.4).

$$\eta_P = \frac{\eta}{P}, \quad P < 1; \tag{2.3}$$

$$\frac{1}{\gamma_P} - 1 = \left(\frac{1}{\gamma} - 1\right)\frac{1}{P^2}, \quad P < 1; \tag{2.4}$$

Figure 2 plots the relation of γ vs. P for each given γ_P in Formula (2.4). When γ_P is less than a predefined maximal threshold (e.g., 2%, 5% or 10%), the two values can be deemed uncorrelated to each other at the level of uncertainty. For each independence standard γ_P, there is a maximal allowed correlation between two values below which the uncorrelated uncertainty assumption of variance arithmetic holds. The maximal allowed correlation is a function of the larger precision of the two values according to Formula (2.4). Figure 2 shows that for two precisely measured values, their correlation γ is allowed to be quite high. Thus, the uncorrelated uncertainty assumption has much weaker statistical requirement than the assumption for independence arithmetic, which requires the two values to be independent of each other.

It is tempting to add noise to otherwise unqualified values to make them satisfy the uncertainties uncertainty assumption. As an extreme case of this approach, if two values are constructed by adding noise to the same signal, they are 50% correlated

at the uncertainty level so that they will not satisfy the uncorrelated uncertainty assumption, which defines the upper bound for γ_P. [4]

2.3 Addition and Subtraction [1]

The Lyapunov form of the central limit theorem [23] states that if X_i is a random variable with mean μ_i and variance V_i for each i among a series of n mutually independent random variables, then with increased n, the sum $\sum_i^n X_i$ converges in distribution to the Gaussian distribution with mean $\sum_i^n \mu_i$ and variance $\sum_i^n V_i$. In the context of the uncorrelated uncertainty assumption, uncertainty in general is expected to be Gaussian distributed [2] [23].

Even if rounding error is the sole source of uncertainty:

- The rounding error of rounding to nearest is shown to be uniformly distributed between the half bit of the least significant bit of the siginficand [1], until the significand itself becomes close to 0.
- The central limit theorem converges the result distribution of addition and subtraction to a Gaussian distribution [1].
- Such convergence to Gaussian distribution is very quick, e.g., after 4 rounds of addition and subtraction, the result distribution is already close to the corresponding Gaussian distribution with the same mean and deviation [1].
- The application of center limit theorem to rounding error is not limited to addition and subtraction, because a multiplication operation contains multiple additions, while a division operation contains multiple subtractions [8]. Thus rounding error is Gaussian distributed in practice.

For simplicity, define operator $\delta^2 f(x) \equiv (\delta f(x))^2$. Formula (2.5) gives the result variance of addition and subtraction surrounding $x \pm y$. Formula (2.6) gives the probability density function ρ for $x \pm \delta x$ in general, in which $N(x)$ is the density function of the Normal distribution [23]. Formula (2.6) can be normalized as Formula (2.7).

$$\delta^2(x \pm y) = (\delta x)^2 + (\delta y)^2; \tag{2.5}$$

$$\rho(\tilde{x}, x, \delta x) = \frac{1}{\delta x} N(\frac{\tilde{x} - x}{\delta x}); \tag{2.6}$$

$$\tilde{z} \equiv \frac{\tilde{x} - x}{\delta x} : \quad \rho(\tilde{x}, x, \delta x) d\tilde{x} = N(\tilde{z}) d\tilde{z}; \tag{2.7}$$

2.4 A Variance Representation

A variance arithmetic representation uses a pair of 64-bit standard floating-point numbers to represent the value x and the variance $(\delta x)^2$ of an imprecise value $x \pm \delta x$. Including the hidden bit, each 64-bit floating-point value has a 53-bit significand [5]. When the uncertainty δx is not specified initially:

[4] The 50% curve in Figure 2 defines the maximal possible correlations between any two measured signals. This other conclusion of Formula (2.4) makes sense because the measurable correlation between two measurements should be limited by the precision of their measurements.

[5] The significand is frequently referred as mantissa.

- For a 64-bit standard floating-point value:
  - If its least 22 bits are all 0, the value is deemed to represent a 2's fractional so that it has 0 as its uncertainty [6];
  - Otherwise, it is assumed to be imprecise in the least significant bit of its 53-bit significand, and the error is assumed uniformly distributed within the bit.
- For an integer value:
  - If it is within the range of $[-2^{53}+1, +2^{53}-1]$, it has 0 as its uncertainty;
  - Otherwise, it is first converted to a conventional floating-point value before converted to an imprecise value.

The equivalent floating-point value of the least significant bit of significand of a floating-point value is defined as the *least significant value* [7], so that the uncertainty is $1/\sqrt{3}$-fold of the least significant value. The term least significant value can be abbreviated as LSV for simplicity.

2.5 Comparison

Two imprecise values can be compared statistically using their difference.

When the value difference is zero, the two imprecise values are equal. In statistics, such two imprecise values have 50% possibility to be less than or greater to each other but no chance to be equal to each other [23]. In variance arithmetic, these two imprecise values are neither less nor greater than each other, so they are equal. In reality, the probability of two values equal exactly to each other is very low.

Otherwise, the standard z-method [23] is used to judge whether they are equal, or less or more than each other. For example, the difference between 1.002 ± 0.001 and 1.000 ± 0.002 is 0.002 ± 0.00224, so that the z-value for the difference is $z = 0.002/0.00224$, and the probability for them to be not equal is $2\xi(|z|) = 62.8\%$, in which $\xi(z)$ is the cumulative density function for Normal distribution [23]. If the threshold probability of not equality is 50%, $1.000 \pm 0.002 < 1.002 \pm 0.001$. Instead of using the threshold probability, the equivalent bounding range for z can be used, such as $|z| \leq 0.67448975$ for the equal threshold probability of 50%.

2.6 Multiplication

The result variance of multiplying $x \pm \delta x$ by a precise value y is $y^2\delta^2 x$ [23]. The result of multiplying $0 \pm \delta x$ by $0 \pm \delta y$ is a normal production distribution [23], which is centered at 0 with variance $(\delta x)^2(\delta y)^2$. The general multiplication can be decomposed as Formula (2.8), which leads to Formula (2.9) [1].

$$(x \pm \delta x) \times (y \pm \delta y) = (x + (0 \pm \delta x)) \times (y + (0 \pm \delta y)); \tag{2.8}$$

$$\delta^2(xy) = x^2(\delta y)^2 + y^2(\delta x)^2 + (\delta x)^2(\delta y)^2; \tag{2.9}$$

Formula (2.9) is identical to Formula (1.12) for statistical propagation of uncertainty except their cross term, representing difference in their statistical requirements,

[6] The possibility of such a value not representing a 2's fractional is less than $2.384\ 10^{-7}$.

[7] The "least significant value" and "unit in the last place" are different in concept.

respectively. Formula (2.9) is simpler and more elegant than Formula (1.5) for interval arithmetic multiplication. The result of Formula (1.2) calculated by variance arithmetic is $2 \pm 2\sqrt{5}$. It is 2 ± 9 for interval arithmetic.

2.7 Distributional Zero and Distributional Pole

Let $\tilde{y} = f(\tilde{x})$ be a strictly monotonic function, so that $\tilde{x} = f^{-1}(\tilde{y})$ exist. Formula (2.10) is the uncertainty distribution density function [2] [23]. In Formula (2.10), the same distribution can be expressed in either $\tilde{x}$ or $\tilde{y}$ or $\tilde{z}$, which are different representations of the same underlying random variable. Using Formula (2.10), Formula (2.11), (2.12), and (2.13) give the $\rho(\tilde{y}, y, \delta y)$ for e^x, $\ln(x)$, and x^c, respectively.

$$N(\tilde{z})d\tilde{z} = \rho(\tilde{x}, x, \delta x)d\tilde{x} = \rho(f^{-1}(\tilde{y}), x, \delta x)\frac{d\tilde{x}}{d\tilde{y}}d\tilde{y} = \rho(\tilde{y}, y, \delta y)d\tilde{y}; \tag{2.10}$$

$$y = e^x : \rho(\tilde{y}, y, \delta y) = \frac{1}{\tilde{y}}\frac{1}{\delta x}N(\frac{\ln(\tilde{y}) - x}{\delta x}); \tag{2.11}$$

$$y = \ln(x) : \rho(\tilde{y}, y, \delta y) = e^{\tilde{y}}\frac{1}{\delta x}N(\frac{e^{\tilde{y}} - x}{\delta x}); \tag{2.12}$$

$$y = x^c : \rho(\tilde{y}, y, \delta y) = c\tilde{y}^{\frac{1}{c}-1}\frac{1}{\delta x}N(\frac{\tilde{y}^{\frac{1}{c}} - x}{\delta x}); \tag{2.13}$$

Viewed in the $f^{-1}(\tilde{y})$ coordinate, $\rho(\tilde{y}, y, \delta y)$ is Gaussian modulated by $\frac{d\tilde{x}}{d\tilde{y}} = 1/f_x^{(1)}$, in which $f_x^{(1)}$ is the first derivative of $f(x)$ to x. A *distributional pole* of the uncertainty distribution happens when $f_x^{(1)} = \infty \rightarrow \rho(\tilde{y}, y, \delta y) = 0$, while a *distributional zero* happens when $f_x^{(1)} = 0 \rightarrow \rho(\tilde{y}, y, \delta y) = \infty$. Zeros and poles have different impacts on the properties of of the uncertainty distributions.

- If $y = f(x)$ is a linear transform of x, $f_x^{(1)}$ is a constant, and $\rho(\tilde{y}, y, \delta y)$ is known to be Gaussian [23]. From another perspective, a linear transformation generates neither distributional zero nor distributional pole according to Formula (2.10).
- Figure 3 shows the probability density function for $(x \pm 1)^2$ according to Formula (2.13), which has a distributional zero at $x = 0$ that separates two defined region $x \geq 0$ and $x \leq 0$. $(0 \pm 1)^2$ is the χ^2 distribution [2]. At the distributional pole, the probability density function resembles a Delta distribution.
- Figure 4 shows the probability density function for $\sqrt{x \pm 1}$ according to Formula (2.13), which has a distributional pole at $x = 0$. At the distributional pole, the probability density function is clamped at zero.

In all the figures, the probability density function in the $\tilde{z}$ representation becomes more Normal-like when the mode of the distributions is further away from either distributional zero or distributional pole.

Formula (2.10) gives the uncertainty distribution of an analytic function. However, normal scientific and engineering calculations usually do not care about the result distribution, but just some simple statistics of the result, such as the result mean and deviation [2] [3]. When the underlying distribution can be simplified as Gaussian distribution, such simple statistics can be obtained using Taylor expansion.

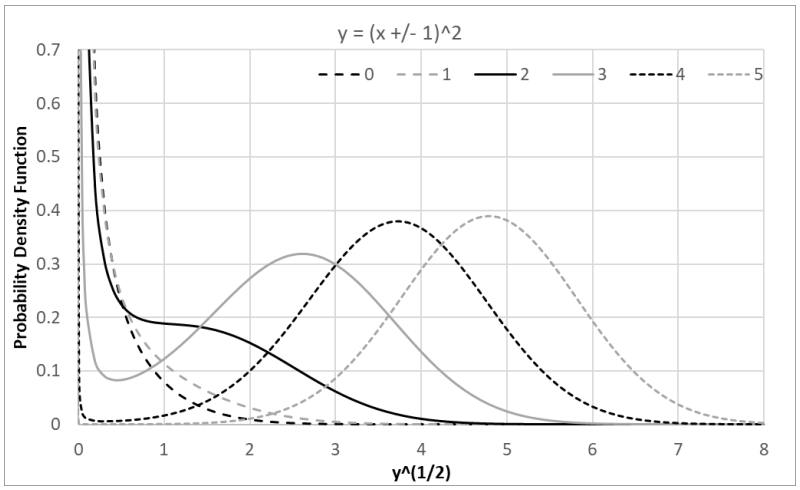

Figure 3: The probability density function for $(x \pm 1)^2$, for different x as shown in the legend. The y-axis is scaled as $\sqrt{\tilde{y}}$. Some probability density function $\rho(\tilde{x})$ in solid line is compared with a Normal distribution $N(\tilde{x})$ of the same mode in the dash line of the same color. The legend for the Normal distribution shows the value of Normal difference in percentage.

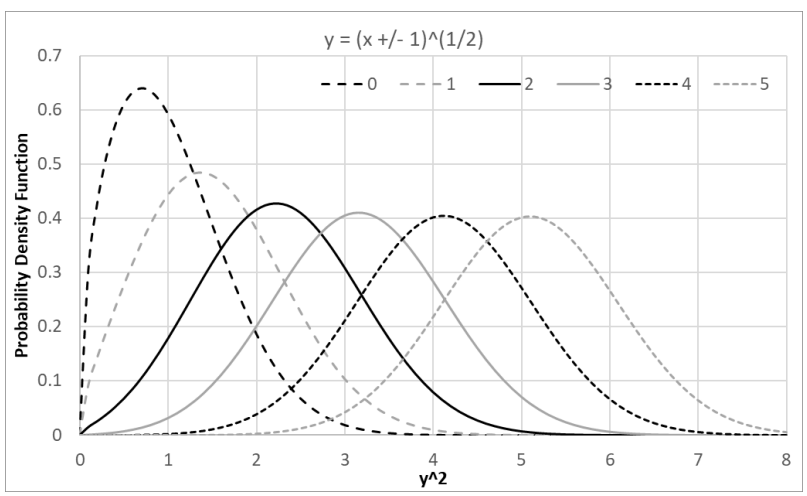

Figure 4: The probability density function for $\tilde{y} = \sqrt{x \pm 1}$, for different x as shown in the legend. The x-axis is scaled as $\tilde{y}^2$. Each probability density function $\rho(\tilde{x})$ in solid line is compared with a Normal distribution $N(\tilde{x})$ of the same mode in the dash line of the same color. The legend for the Normal distribution shows the value of Normal difference in percentage.

2.8 Analytic Functions

2.8.1 Uncertainty of Taylor Expansion

An analytic function $f(x)$ can be precisely calculated in a range by the Taylor series as shown in Formula (2.14). Using Formula (2.7), Formula (2.17) and (2.18) [1] gives the mean $\overline{f(x)}$ and the variance $\delta^2 f(x)$, respectively, in which $\zeta(n)$ is the *variance momentum* defined by Formula (2.15), and σ is defined as the *bounding factor*. Formula (2.16) holds for any symmetric uncertainty distribution around its mean. Formula (2.17) is deduced from Formula (2.14) by $\tilde{x}^{2n} \to \zeta(2n)(\delta x)^{2n}$ and $\tilde{x}^{2n+1} \to 0$. Formula (2.18) is similarly deducted from the square of Formula (2.14). For simplicity, only Gaussian input uncertainty distribution is discussed in this paper, although Formula (2.17) and (2.18) to other input uncertainty distributions.

$$\tilde{y} = f(x + \tilde{x}) = f(x + \tilde{z}\delta x) = \sum_{n=0}^{\infty} \frac{f_x^{(n)}}{n!} \tilde{z}^n (\delta x)^n; \tag{2.14}$$

$$\zeta(n) \equiv \int_{-\sigma}^{+\sigma} \tilde{z}^n N(\tilde{z}) d\tilde{z}; \tag{2.15}$$

$$\zeta(2n+1) = 0; \tag{2.16}$$

$$\overline{f(x)} = \int_{-\sigma}^{+\sigma} f(x + \tilde{z}\delta x) N(\tilde{z}) d\tilde{z} = f(x) + \sum_{n=1}^{\infty} (\delta x)^{2n} \frac{f_x^{(2n)} \zeta(2n)}{(2n)!}; \tag{2.17}$$

$$\delta^2 f(x) = \overline{f(x)^2} - \overline{f(x)}^2 = \sum_{n=0}^{\infty} (\delta x)^{2n} \tag{2.18}$$

$$\left(\zeta(2n) \sum_{j=1}^{2n-1} \frac{f_x^{(j)}}{j!} \frac{f_x^{(2n-j)}}{(2n-j)!} - \sum_{j=1}^{n-1} \frac{f_x^{(2j)} \zeta(2j)}{(2j)!} \frac{f_x^{(2n-2j)} \zeta(2n-2j)}{(2n-2j)!} \right);$$

$\overline{f(x)} - f(x)$ is defined as the *uncertainty bias* which is the effect of the input uncertainty to the result, and which is usually ignored in the conventional floating-point calculations.

2.8.2 Bounding Factor and Bounding Leakage

Formula (2.15) provides statistical bounding $\tilde{x} \in (x - \sigma\delta x, x + \sigma\delta x)$. If $\sigma = \infty$, $\zeta(2n) = (2n-1)!!$, which causes Formula (2.18) to diverge for $\log(x \pm \delta x)$, and $(x \pm \delta x)^c, c \in R$. In another word, statistical bounding is a necessity. When σ is limited, Formula (2.15) becomes Formula (2.19):

- For small $2n$, $\zeta(2n)$ can be approximated by $\zeta(2n) = (2n-1)!!$ according to Formula (2.20). When $\sigma = 5, n < 5$, $|1 - \eta(2n)/(2n-1)!!| < 10^{-3}$.
- For large $2n$, Formula (2.19) reduces to Formula (2.21), which shows that $\zeta(2n)$ increases slower than σ^{2n} for increasing $2n$.

$$\zeta(2n) = 2N(\sigma)\sigma^{2n} \sum_{j=1}^{\infty} \sigma^{2j-1} \frac{(2n-1)!!}{(2n-1+2j)!!}; \tag{2.19}$$

$$= (2n-1)\zeta(\sigma, 2n-2) - 2N(\sigma)\sigma^{2n-1}; \tag{2.20}$$

$$\sigma^2 \ll 2n: \quad \zeta(2n) \simeq 2N(\sigma) \frac{\sigma^{2n+1}}{2n+1}; \tag{2.21}$$

The property of $\zeta(2n)$ when $\sigma^2 \ll 2n$ is not sensitive to the input uncertainty distribution. If the Normal distribution $N(\tilde{z})$ in Formula (2.15) is replaced by a uniform distribution between $[-\sigma, +\sigma]$, Formula (2.22) is similar to Formula (2.21):

$$\zeta(2n) \equiv \int_{-\sigma}^{+\sigma} \frac{1}{2\sigma} \tilde{z}^{2n} d\tilde{z} = 2\frac{1}{2\sigma}\frac{\sigma^{2n+1}}{2n+1}; \tag{2.22}$$

The limited range of $\tilde{x} \in (-\sigma\delta x, +\sigma\delta x)$ in Formula (2.15) causes a bounding leakage ϵ according to Formula (2.23), in which $\xi(z)$ is the cumulative density function for Normal distribution [23]. When $\sigma = 5$, $\epsilon = 5.73 \times 10^{-7}$, which is small enough for most applications. $\sigma = 5$ is also a golden standard to assert a negative result [3].

$$\epsilon = 2 - 2\xi(\frac{1}{2} + \sigma); \tag{2.23}$$

Thus, $\sigma = 5$ in variance arithmetic.

2.8.3 One-dimensional Examples

Formula (2.25) and (2.26) give the uncertainty bias and variance for e^x, respectively:

$$e^{x+\tilde{x}} = e^x \sum_{n=0}^{\infty} \frac{\tilde{x}^n}{n!}; \tag{2.24}$$

$$\frac{\overline{e^x}}{e^x} = 1 + \sum_{n=1}^{\infty} (\delta x)^{2n} \zeta(2n) \frac{1}{(2n)!}; \tag{2.25}$$

$$\frac{\delta^2 e^x}{(e^x)^2} = \sum_{n=1}^{\infty} (\delta x)^{2n} \left(\sum_{j=1}^{2n-1} \frac{\zeta(2n)}{j!\,(2n-j)!} - \sum_{j=1}^{n-1} \frac{\zeta(2j)}{(2j)!}\frac{\zeta(2n-2j)}{(2n-2j)!} \right) \tag{2.26}$$

$$\simeq (\delta x)^2 + \frac{3}{2}(\delta x)^4 + \frac{7}{6}(\delta x)^6 + \frac{5}{8}(\delta x)^8 + o((\delta x)^{10}); \tag{2.27}$$

Formula (2.29) and (2.30) give the uncertainty bias and variance for $\ln(x)$, respectively:

$$\ln(x + \tilde{x}) - \ln(x) = \ln(1 + \frac{\tilde{x}}{x}) = \sum_{j=1}^{\infty} \frac{(-1)^{j+1}}{j} \frac{\tilde{x}^j}{x^j}; \tag{2.28}$$

$$\overline{\ln(x)} = \ln(x) - \sum_{n=1}^{+\infty} P(x)^{2n} \frac{\zeta(2n)}{2n}; \tag{2.29}$$

$$\delta^2 \ln(x) = \sum_{n=1}^{+\infty} P(x)^{2n} \left(\sum_{j=1}^{2n-1} \frac{\zeta(2n)}{j(2n-j)} - \sum_{j=1}^{n-1} \frac{\zeta(2j)}{2j}\frac{\zeta(2n-2j)}{2n-2j} \right) \tag{2.30}$$

$$\simeq P(x)^2 + P(x)^4 \frac{9}{8} + P(x)^6 \frac{119}{24} + P(x)^8 \frac{991}{32} + o(P(x)^{10}); \tag{2.31}$$

Formula (2.33) and (2.34) give the uncertainty bias and variance for $\sin(x)$, re-

spectively:

$$\sin(x+\tilde{x}) = \sum_{n=0}^{\infty} \sin(x)(-1)^n \frac{\tilde{x}^{2n}}{(2n)!} + \sum_{n=0}^{\infty} \cos(x)(-1)^n \frac{\tilde{x}^{2n+1}}{(2n+1)!}; \tag{2.32}$$

$$\overline{\sin(x)} = \sin(x) + \sin(x) \sum_{n=0}^{\infty} (\delta x)^{2n} (-1)^n \frac{\zeta(2n)}{(2n)!}; \tag{2.33}$$

$$\delta^2 \sin(x) = \sum_{n=1}^{\infty} (\delta x)^{2n} (-1)^{n-1} \tag{2.34}$$

$$\left(\cos(x)^2 \sum_{j=0}^{n-1} \frac{\zeta(2n)}{(2j+1)!(2n-2j-1)!} - \sin(x)^2 \sum_{j=1}^{n-1} \frac{\zeta(2n) - \zeta(2j)\zeta(2n-2j)}{(2j)!(2n-2j)!} \right)$$
$$\simeq (\delta x)^2 \cos(x)^2 - (\delta x)^4 (\cos(x)^2 \frac{3}{2} - \frac{1}{2}) + (\delta x)^6 (\cos(x)^2 \frac{7}{6} - \frac{1}{2}) + o((\delta x)^8); \tag{2.35}$$

Formula (2.37) and (2.38) give the uncertainty bias and variance for x^c, respectively, in which $\binom{c}{n} = \frac{\prod_{j=0}^{n-1}(c-j)}{n!}$:

$$(x+\tilde{x})^c = x^c (1 + \frac{\tilde{x}}{x})^c = x^c + x^c \sum_{n=1}^{\infty} \frac{\tilde{x}^n}{x^n} \binom{c}{n}; \tag{2.36}$$

$$\frac{\overline{x^c}}{x^c} = 1 + 1 \sum_{n=1}^{\infty} P(x)^{2n} \zeta(2n) \binom{c}{2n}; \tag{2.37}$$

$$\frac{\delta^2 x^c}{(x^c)^2} = \sum_{n=1}^{\infty} P(x)^{2n} \left(\sum_{j=1}^{2n-1} \zeta(2n) \binom{c}{j} \binom{c}{2n-j} - \sum_{j=1}^{n-1} \zeta(2j) \binom{c}{2j} \zeta(2n-2j) \binom{c}{2n-2j} \right) \tag{2.38}$$

$$\simeq c^2 P(x)^2 + \frac{3}{2} c^2 (c-1)(c - \frac{5}{3}) P(x)^4 + \frac{7}{6} c^2 (c-1)(c-2)^2 (c - \frac{16}{7}) P(x)^6 + o(P(x)^8); \tag{2.39}$$

As special cases for Formula (2.37) and (2.38), (2.40), (2.41), (2.42), (2.43), (2.44), (2.45), (2.46), and (2.47) give the mean and variance for x^2, $\sqrt{x}$, $1/x$, and $1/x^2$, respectively:

$$\overline{x^2} \simeq x^2 + (\delta x)^2; \tag{2.40}$$

$$\delta^2 x^2 \simeq 4x^2 (\delta x)^2 + 2(\delta x)^4; \tag{2.41}$$

$$\frac{\overline{\sqrt{x}}}{\sqrt{x}} \simeq 1 - \frac{1}{8} P(x)^2 - \frac{15}{128} P(x)^4 - \frac{315}{1024} P(x)^6 + o(P(x)^8); \tag{2.42}$$

$$\frac{\delta^2 \sqrt{x}}{(\sqrt{x})^2} \simeq \frac{1}{4} P(x)^2 + \frac{7}{32} P(x)^4 + \frac{75}{128} P(x)^6 + o(P(x)^8); \tag{2.43}$$

$$\frac{\overline{1/x}}{1/x} \simeq 1 + P(x)^2 + 3P(x)^4 + 15P(x)^6 + o(P(x)^8); \tag{2.44}$$

$$\frac{\delta^2 1/x}{(1/x)^2} \simeq P(x)^2 + 8P(x)^4 + 69P(x)^6 + o(P(x)^8); \tag{2.45}$$

$$\frac{\overline{1/x^2}}{1/x^2} \simeq 3P(x)^2 + 15P(x)^4 + 105P(x)^6 + o(P(x)^8); \tag{2.46}$$

$$\frac{\delta^2 1/x^2}{(1/x^2)^2} \simeq 4P(x)^2 + 66P(x)^4 + 960P(x)^6 + o(P(x)^8); \tag{2.47}$$

The variance formulas using Taylor expansion show the nature of the calculation, such as $\delta x \to P(e^x)$, $\delta x \to \delta \sin(x)$, $P(x) \to P(x^c)$, and $P(x) \to \delta \ln(x)$, and the difference speed of variance increases of x^c depending on different c. Section 7 will discuss the convergence of Formula (2.26), (2.30), (2.34), and (2.38), as well as the statistical context for the convergence in details.

2.8.4 Multiple Dimensional Expansion

Due to uncorrelated uncertainty assumption, the Taylor expansion can be applied to multiple inputs, such as Formula (2.50) [1] .

$$f(x+\tilde{x}, y+\tilde{y}) = \sum_{m=0}^{\infty}\sum_{n=0}^{\infty} \frac{f_{(x,y)}^{(m,n)}}{m!n!}\tilde{x}^m\tilde{y}^n; \tag{2.48}$$

$$\begin{aligned}\overline{f(x,y)} &= \int\int f(x+\tilde{x}, y+\tilde{y})\rho(\tilde{x}, x, \delta x)\rho(\tilde{y}, y, \delta y)\, d\tilde{x}d\tilde{y} \\ &= \sum_{m=0}^{\infty}\sum_{n=0}^{\infty}(\delta x)^{2m}(\delta y)^{2n}\frac{\zeta(2m)\zeta(2n)f_{(x,y)}^{(2m,2n)}}{(2m)!(2n)!};\end{aligned} \tag{2.49}$$

$$\begin{aligned}\delta^2 f(x,y) = \overline{f(x,y)^2} - \overline{f(x,y)}^2 &= \sum_{m=0}^{\infty}\sum_{n=0}^{\infty}(\delta x)^{2m}(\delta y)^{2n} \\ &(\zeta(2m)\zeta(2n)\sum_{i=0}^{2m}\sum_{j=0}^{2n}\frac{f_{(x,y)}^{(i,j)}}{i!\,j!}\frac{f_{(x,y)}^{(2m-i,2n-j)}}{(2m-i)!\,(2n-j)!} \\ &-\sum_{i=0}^{m}\sum_{j=0}^{n}\frac{\zeta(2i)\zeta(2j)f_{(x,y)}^{(2i,2j)}}{(2i)!(2j)!}\frac{\zeta(2m-2i)\zeta(2n-2j)f_{(x,y)}^{(2m-2i,2n-2j)}}{(2m-2i)!(2n-2j)!});\end{aligned} \tag{2.50}$$

Formula (2.5) and (2.9) are two special cases of Formula (2.50). Because the Taylor expansions for x/y and $x\ 1/y$ are identical, $x/y = x\ 1/y$.

Although Formula (2.50) is only for 2-dimensional, it can be extended to any dimensional.

2.9 Dependency Tracing

$$\delta^2(f+g) = \delta^2 f + \delta^2 g + 2(\overline{fg} - \overline{f}\,\overline{g}); \tag{2.51}$$

$$\delta^2(fg) = \overline{f^2g^2} - \overline{fg}^2; \tag{2.52}$$

$$\delta^2 f(g) = \overline{f(g)^2} - \overline{f(g)}^2; \tag{2.53}$$

$$\delta^2(c_1 f + c_0) = c_1^2\delta^2 f; \tag{2.54}$$

When all the inputs obey the uncorrelated uncertainty assumption, variance arithmetic uses statistics to trace dependency in the intermediate steps. By convention, variance arithmetic treats different input variables as imprecise values satisfying uncorrelated uncertainty assumption. For example:

Table 1: The dependency problem when applying variance arithmetic without dependency tracing for $x^2 - x$ when the input variance is $(\delta x)^2$. The correct result variance is $(2x-1)^2(\delta x)^2 + 2(\delta x)^4$ which corresponds to (1.7).

Formula	Result difference	Wrong independence
(1.8)	$4x(\delta x)^2$	between x^2 and x
(1.9)	$-(2x^2-2x)(\delta x)^2 - (\delta x)^4$	between $x-1$ and x

- Formula (2.51) shows $\delta^2(f+g)$, whose dependence tracing can be demonstrated by $\delta^2(f-f) = 0$, and $\delta^2(f(x)+g(y)) = \delta^2 f + \delta^2 g$, with the latter case corresponding to Formula (2.5). Formula (2.55) shows that Formula (2.18) uses Formula (2.51) between any two terms in the Taylor expansion of Formula (2.14).
- Formula (2.52) shows $\delta^2(fg)$, whose dependence tracing can be demonstrated by $\delta^2(f/f) = 0$, $\delta^2(ff) = \delta^2 f^2$, and $\delta^2(f(x)g(y)) = \overline{f}^2(\delta^2 g) + (\delta^2 f)\overline{g}^2 + (\delta^2 f)(\delta^2 g)$, with the latter case corresponding to Formula (2.9).
- Formula (2.53) shows $\delta^2 f(g(x))$, whose dependence tracing can be demonstrated by $\delta^2(f^{-1}(f)) = (\delta x)^2$.
- Formula (2.54) shows the variance of linear transformation of a function, which can be applied to Formula (2.51) and (2.52) for more generic dependency tracing.

$$\delta^2\left(\frac{f_x^{(m)}\tilde{x}^m}{m!} + \frac{f_x^{(n)}\tilde{x}^n}{n!}\right) = (\delta x)^{2m}(\frac{f_x^{(m)}}{m!})^2\eta(2m) + (\delta x)^{2n}(\frac{f_x^{(n)}}{n!})^2\eta(2n) \tag{2.55}$$
$$+ 2(\delta x)^{m+n}\left(\frac{f_x^{(m)}f_x^{(n)}}{m!\,n!}\eta(m+n) - \frac{f_x^{(m)}}{m!}\eta(m)\,\frac{f_x^{(n)}}{n!}\eta(n)\right);$$

Variance arithmetic uses dependency tracing to guarantee that the result uncertainty bias and uncertainty fit statistics. Dependency trading comes with a price: variance calculation is generally much more complex than value calculation.

2.10 Traditional Execution and Dependency Problem

Dependency tracing requires an analytic solution to apply Taylor expansion for the result mean and variance, such as using Formula (2.18) and (2.50). It conflicts with traditional numerical executions on analytic functions:

- Traditionally, using intermediate variables is a common practice. But it upsets dependency tracing through variable usage.
- Traditionally, conditional executions are used to optimize numerical executions and to minimize rounding errors, such as the Gaussian elimination to minimize floating-point rounding errors for matrix inversion [36]. Unless required by mathematics such as for the definition of identity matrix, conditional executions need to be removed for dependency tracing, e.g., Gaussian elimination needs to be replaced by direct matrix inversion as described in Section 5.
- Traditionally, approximations on the result values are carried out during executions. Using variance arithmetic, Formula (2.18) shows that the variance converges much slower than that of the value in Taylor expansion, so that the target of approximation should be more on the variance. Section 5 provides an example of first order approximation on calculating matrix determinant.

- Traditionally, results from math library functions are trusted blindly down to every bit. As shown in Section 9, variance arithmetic can now illuminate the numerical errors in math library functions, and demands these functions to be recalculated with uncertainty as part of the output.
- Traditionally, an analytic expression can be broken down into simpler and independent arithmetic operations of negation, addition, multiplication, and division [8]. But this is no longer true for uncertainty-bearing arithmetic. For example, if $x^2 - x$ is calculated as Formula (1.7), (1.8), and (1.9) using Formula (2.5), (2.9), and (2.41) as separated and independent arithmetic operations, only Formula (1.7) gives the correct result because the input variable x appears only once, while the other two give wrong results for wrong independence assumptions, respectively, as shown in Table 1. In variance arithmetic, Taylor expansion unifies different analytic expressions of the same function, to give the same result [9].
- Traditionally, a large calculation may be broken into sequential steps, such as calculating $f(g(x))$ as $f(y)|_{y=g(x)}$. But this method generally no longer works in variance arithmetic due to dependency tracing of $g(x)$ inside $f(g(x))$. According to Formula (2.56), (2.57), (2.58), and (2.59), in variance arithmetic, $f(g(x)) \neq f(y)|_{y=g(x)}$, with the latter suffers from the dependency problem according to Formula (2.60), (2.61), (2.62) and (2.63).

$$\overline{f(g(x))} - f(g(x)) = o((\delta x)^4) + (\delta x)^2 \frac{1}{2}\left(f_x^{(2)}(g_x^{(1)})^2 + f_x^{(1)} g_x^{(2)}\right); \tag{2.56}$$

$$\overline{f(g)|_{g(x)}} - f(g(x)) = o((\delta g)^4) + (\delta x)^2 \frac{1}{2} f_x^{(2)}(g_x^{(1)})^2; \tag{2.57}$$

$$\begin{aligned}\delta^2 f(g(x)) = {} & o((\delta x)^6) + (\delta x)^2 (f_x^{(1)} g_x^{(1)})^2 + \\ & (\delta x)^4 \Big(f_x^{(1)} f_x^{(3)} (g_x^{(1)})^4 + 4 f_x^{(1)} f_x^{(2)} (g_x^{(1)})^2 g_x^{(2)} + (f_x^{(1)})^2 g_x^{(1)} g_x^{(3)} + \\ & \frac{1}{2}(f_x^{(2)})^2 (g_x^{(1)})^4 + \frac{1}{2}(f_x^{(1)})^2 (g_x^{(2)})^2\Big);\end{aligned} \tag{2.58}$$

$$\begin{aligned}\delta^2 f(g))|_{g(x)} = {} & o((\delta x)^6) + (\delta x)^2 (f_x^{(1)} g_x^{(1)})^2 + \\ & (\delta x)^4 \left(f_x^{(1)} f_x^{(3)} (g_x^{(1)})^2 + \frac{1}{2}(f_x^{(2)})^2 (g_x^{(1)})^2\right);\end{aligned} \tag{2.59}$$

$$\overline{(\sqrt{x})^2} - x = o((\delta x)^4) + \frac{1}{4}(\delta x)^2; \tag{2.60}$$

$$\overline{\sqrt{x^2}} - x = o((\delta x)^4) - \frac{1}{2}(\delta x)^2; \tag{2.61}$$

$$\delta^2 y^2|_{y=\sqrt{x}} = o((\delta x)^6) + (\delta x)^2 + 4(\delta x)^4; \tag{2.62}$$

$$\delta^2 \sqrt{y}|_{y=x^2} = o((\delta x)^6)(\delta x)^2 + (\delta x)^4; \tag{2.63}$$

When the dependency tracing is not applied strictly, variance arithmetic will have dependency problem, as demonstrated by Formula (2.60), (2.60), (2.62), and (2.63). Dependency tracing gets rid of almost all freedoms in the traditional numerical executions, as well as the associated dependence problems. All traditional numerical algorithms need to be reexamined or reinvented for variance arithmetic.

[8] Sometimes square root is also regarded as an arithmetic operation.

[9] Until now, Taylor expansion cannot be used as a general method in interval arithmetic.

3 Verification of Variance arithmetic

3.1 Verification Methods and Standards

Analytic functions or algorithms each with precisely known results are used to validate the result from variance arithmetic using the following statistical properties:

- *value error*: as the difference between the numerical result and the corresponding known precise analytic result.
- *normalized error*: as the ratio of a value error to the corresponding uncertainty.
- *error deviation*: as the standard deviation of the normalized errors.
- *error distribution*: as the histogram of the normalized errors.
- *uncertainty mean*: as the mean of the result uncertainties.
- *uncertainty deviation*: as the deviation of the result uncertainties.
- *error response*: as the relation between the input uncertainties and the result uncertainties.
- *calculation response*: as the relation between the amount of calculation and the result uncertainties.

One goal of calculation involving uncertainty is to precisely account for all input errors from all sources, to achieve *ideal coverage*. In the case of ideal coverage:

1. The error deviation is exactly 1.
2. The result error distribution should be Normal distribution when an imprecise value is expected, or Delta distribution when a precise value such as a true zero is expected.
3. The error response fits the function, such as a linear error response is expected for a linear function.
4. The calculation response fits the function, such as more calculations generally result in larger result uncertainties.

If the precise result is not known, according to Section 5, the result uncertainty distribution can be used to judge whether ideal coverage is achieved or not.

If instead, the input uncertainty is only correct for the input errors on the order of magnitude, the *proper coverage* is achieved when the error deviations are in the range of $[0.1, 10]$.

When an input contains unspecified errors, such as floating-point rounding errors, or numerical errors of library functions, Gaussian or white noises of increasing deviations can be added to the input, until ideal coverage is reached. The needed minimal noise deviation gives a good estimation of the amount of unspecified input uncertainties. The existence of ideal coverage is a necessary verification if the application of Formula (2.18) or (2.50) has been applied correctly for the application. The input noise range for the idea coverage defines the ideal application range of input uncertainties.

Gaussian or white noises of specified deviations can also be added to the input to measure the error responses of a function.

3.2 Types of Uncertainties

There are five sources of result uncertainty for a calculation [2][11]:

- input uncertainties,
- rounding errors,
- truncation errors,
- external errors,
- modeling errors.

The examples in this paper show that when the input uncertainty precision is 10^{-15} or larger, variance arithmetic can provide ideal coverage for input uncertainties.

Empirically, variance arithmetic can provide proper coverage to rounding errors, with the error deviations in the range of $[0.4, 2.5]$.

Section 4 shows that variance arithmetic no longer needs theoretical reminder estimator [11]. Instead, it uses stability truncation, to guarantee that the truncation error is statistically small enough for the result uncertainty.

The *external errors* are the value errors which are not specified by the input uncertainties, such as the numerical errors in the library math functions. Section 9 studies the effect of sin and tan numerical errors [11]. It demonstrates that when the external errors are large enough, neither ideal coverage nor proper coverage can be achieved. It shows that the library functions need to be recalculated to have corresponding uncertainty attached to each value.

The *modelling errors* arise when an approximate analytic solution is used, or when a real problem is simplified to obtain the solution. For example, Section 9 demonstrates that the discrete Fourier transformation is only an approximation for the mathematically defined Fourier transformation. Conceptually, modelling errors originate from mathematics, so they are outside the domain for variance arithmetic.

3.3 Types of Calculations to Verify

Algorithms of completely different nature with each representative for its category are needed to test the generic applicability of variance arithmetic [1]. An algorithm can be categorized by comparing the amount of its input and output data [1]:

- application,
- transformation,
- generation,
- reduction.

An *application* algorithm calculates numerical values from an analytic formula. Through Taylor expansion, variance arithmetic is directly applicable to analytic problems.

A *transformation* algorithm has about equal amounts of input and output data. The information contained in the data remains about the same after transforming. For reversible transformations, a unique requirement for variance arithmetic is to recover the original input uncertainties after a *round-trip* transformation which is a *forward* followed by a *reverse* transformation. Discrete Fourier Transformation is a typical reversible transforming algorithm, which contains the same amount of input and output data, and its output data can be transformed back to the input data using essentially

the same algorithm. A test of variance arithmetic using FFT algorithms is provided in Section 9.

A *generation* algorithm has much more output data than input data. The generating algorithm codes mathematical knowledge into data, so there is an increase of information in the output data. Some generating algorithms are theoretical calculations which involve no imprecise input so that all result uncertainty is due to rounding errors. Section 10 demonstrates such an algorithm, which calculates a table of the sine function using trigonometric relations and two precise input data, $sin(0) = 0$ and $sin(\pi/2) = 1$.

A *reduction* algorithm has much less output data than input data such as numerical integration and statistical characterization of a data set. Some information of the data is lost while other information is extracted during reducing. Conventional wisdom is that a reducing algorithm generally benefits from a larger input data set [23]. A test of statistical characterization through sampling is provided in Section 8.

4 Polynomial and Taylor Expansion

4.1 Polynomial

For a polynomial $f(x) \equiv \sum_{j=0}^{N} c_j x^j$, Formula (4.1) gives the corresponding Taylor expansion:

$$f(x+\tilde{x}) = \sum_{j=0}^{N} c_j (x+\tilde{x})^j = \sum_{j=0}^{N} \tilde{x}^j \sum_{k=0}^{N-j} x^{k-j} c_{j+k} \binom{j+k}{j}; \tag{4.1}$$

When $f(x) = (ax+b)^N$, the result of applying Formula (4.1) to Formula (2.18) is Formula (2.38) with $c = N$ and $x = ax + b$. Unlike Formula (2.38), Formula (4.1) converges for all $x \pm \delta x$, because the exponent for power is limited to positive integers only.

4.2 Truncation Error

Formula (2.55) shows that Taylor expansion such as Formula (2.17) and (2.18) can be treated as polynomial with $N \to \infty$.

A numerical calculation can only calculate limited terms in a Taylor expansion, and the count of terms of the Taylor expansion to approximate a function $f(x)$ is defined as the *Taylor expansion order*. The needed Taylor expansion order can be obtained by theoretical reminder estimation: If the estimation is less than LSV of the current expansion, the Taylor expansion can stop [11]. However, this method has not considered floating-point rounding errors, so that it cannot guarantee that the result is correct until LSV.

Without reminder estimator, the expansion is stopped if the change of the expansion is less than an ad hoc threshold [11]. Such truncation by stability is generally not reliable because traditional calculation has no idea on what the result uncertainty should be. In practice, if the value change of the expansion is less than the LSV of the current expansion, the expansion will stop [11]. This approach incorporates excessive rounding error because the Taylor expansion order is larger than needed.

Formula (4.2) gives the Taylor expansion of $1/(1-x)$. In Formula (4.2), each $|x|^n$ generates the same rounding error, but the difference in the sign of x means that the same rounding error will have different effects for the accumulated rounding error. Instead of a reminder estimator, the true reminder $r(N)$ is calculated as Formula (4.3).

$$e(N) = \sum_{n=0}^{N} x^n = \begin{cases} x > 0: & 1 + |x| + |x|^2 + |x|^3 + \dots \\ x < 0: & 1 - |x| + |x|^2 - |x|^3 + \dots \end{cases}; \tag{4.2}$$

$$r(N) = \left| e(N) - \frac{1}{1-x} \right|; \tag{4.3}$$

By treating LSV of each conventional floating-point value as imprecise, variance arithmetic can trace floating-point rounding errors. Formula (4.2) tests the ability of variance arithmetic to predict the effect of floating-point rounding errors.

4.3 Rounding Error

Figure 5 shows $r(N)$ and the corresponding uncertainties for $x = \pm 0.6, \pm 0.7$. For $x = \pm 0.6, -0.7$, after $r(N)$ reaches LSV of $1/(1-x)$, the LSV equals the corresponding

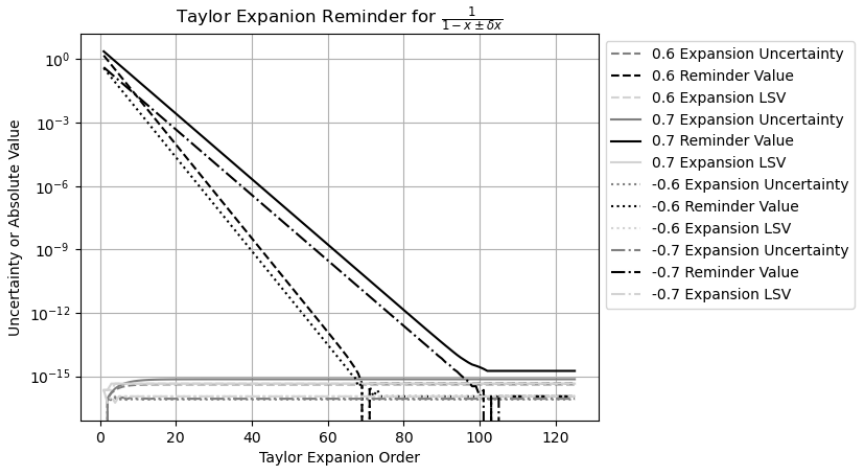

Figure 5: The reminders and result uncertainties for Taylor expansion of $1/(1-x)$, for different Taylor expansion order N as shown in the x-axis, and different x as shown in the legend. The y-axis is for either absolute reminder or expansion uncertainty. In the legend, *Reminder Value* is the value of Formula (4.3), and *Expansion Uncertainty* is the uncertainty of Formula (4.2).

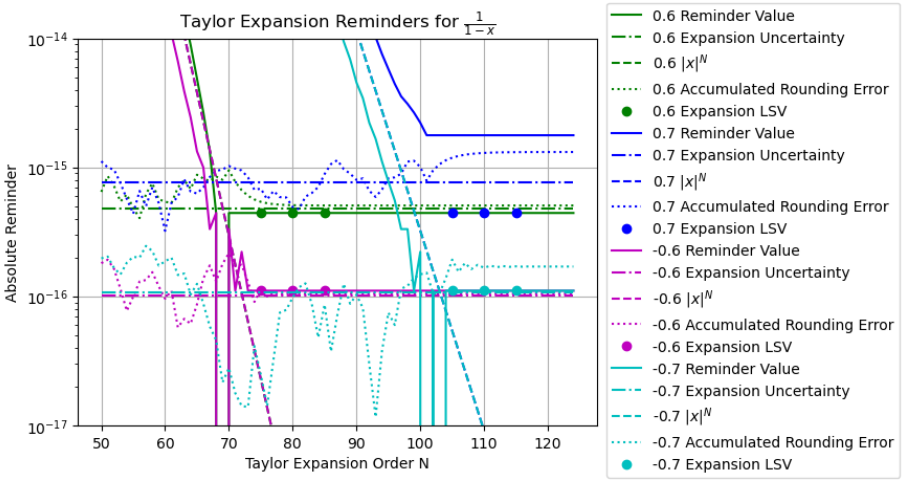

Figure 6: The reminders for Taylor expansion of $1/(1-x)$, for different Taylor expansion order N as shown in the x-axis, and different x as shown in the legend. The y-axis is for absolute reminder. In the legend, *Reminder Value* is the value of Formula (4.3), *Expansion LSV* is the *LSV* of the value of Formula (4.2), and *Accumulated Rounding Error* is the sum of rounding errors of $x^n, n = 1 \dots N$.

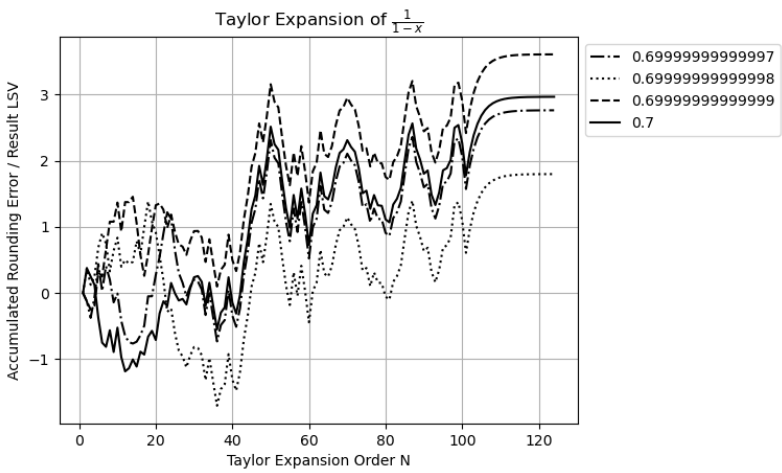

Figure 7: The accumulated rounding errors for Taylor expansion of $1/(1-x)$, for different Taylor expansion order N as shown in the x-axis, and different x as shown in the legend. The y-axis is for accumulated rounding errors.

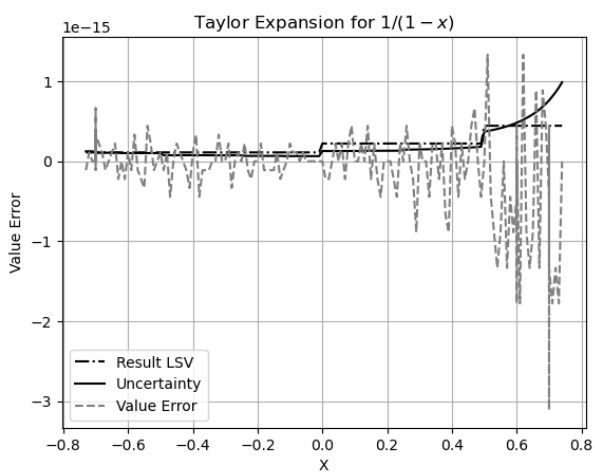

Figure 8: The accumulated rounding errors for Taylor expansion of $1/(1-x)$, for different x as shown in the x-axis. The y-axis is for accumulated rounding errors. In the legend, *Result LSV* is the *LSV* of the value of $1/(1-x)$, *Uncertainty* is the uncertainty of Formula (4.2), and *Value Error* is the value of Formula (4.3).

expansion uncertainty, so that variance arithmetic reproduces the result of traditional expansion truncation. For $x = 0.7$, $r(N)$ does not reach LSV, leaving the least two significant bits incorrect, which shows that traditional numerical Taylor expansion can be less accurate than its theoretical claim.

In addition to $r(N)$, Figure 6 shows expansion LSV, $|x|^N$, and accumulated rounding errors, zooming in at where the expansions terminate.

1. When x^N is added to $e(N)$ in Formula (4.2), it is digitized by the LSV of $e(N)$, to generate rounding error at N. For example, $0.7^{100} = 3.23\ 10^{-16}$ is digitized by the $LSV = 4.44\ 10^{-16}$ as $4.44\ 10^{-16}$ for $1/(1-0.7)$. Such digitization is observed to be round to the nearest, which agrees with the IEEE standard [7].
2. When x^N reaches below half of LSV of $e(N)$, it vanishes so that $e(N)$ reaches its final value.
3. Figure 6 shows that while the accumulated rounding errors fluctuate around 0, the last few accumulated rounding errors when x^N approaches LSV of $e(N)$ determine the final value error of $e(N)$, such as the accumulation of 2-bit errors for $1/(1-0.7)$.
4. When x changes slightly, the rounding errors in Figure 6 retains the same pattern while the value of the expansion changes, suggesting that the rounding errors are not completely random. Thus, variance arithmetic is not expected to trace rounding errors precisely.
5. $1/(1-0.6)$ has similar coherent accumulated rounding errors, but because the corresponding $|x|^N$ decreases faster as shown in Figure 6, the final accumulated error is less than that of $1/(1-0.7)$.
6. The sign difference in Formula (4.2) for $1/(1+0.7)$ determines that the accumulated rounding errors vs Taylor expansion order N have different and less coherent pattern than that of $1/(1-0.7)$.

A value error due to rounding error results from coherent contribution of consecutive rounding errors, so that it is not a random process. Figure 8 shows value errors and uncertainties for $1/(1-x), x \in [-0.75, +0.75]$. It shows that the calculated uncertainties from variance arithmetic provides an average coverage to value errors, resulting in an overall error deviation of 2.22. The step change of uncertainty in Figure 8 is due to change of result LSV. Variance arithmetic can only give proper coverage for floating-point rounding errors.

4.4 Stability Truncation

The input to an expansion can also be imprecise. Figure 9 shows that the absolute reminder values decrease exponentially with increasing expansion order, while the expansion uncertainty stabilizes quickly above the input uncertainty 0.01. The expansion can terminate when the reminder value equals the expansion value LSV for $x = \pm 0.6 \pm 0.01, -0.7 \pm 0.01$, but not for $x = 0.7 \pm 0.01$. Figure 9 is very similar to Figure 5 except that the expansion uncertainty is much larger.

When input uncertainty increases from 0.01 to 0.04, Figure 10 shows that the absolute reminder value decreases slower than exponentially with increasing expansion order, while the expansion uncertainty remains about the same with increasing Taylor expansion order after reaching their stable values, respectively. For example, the expansion of $1/(1 + 0.7 \pm 0.04)$ now terminates at Taylor expansion order 105 instead

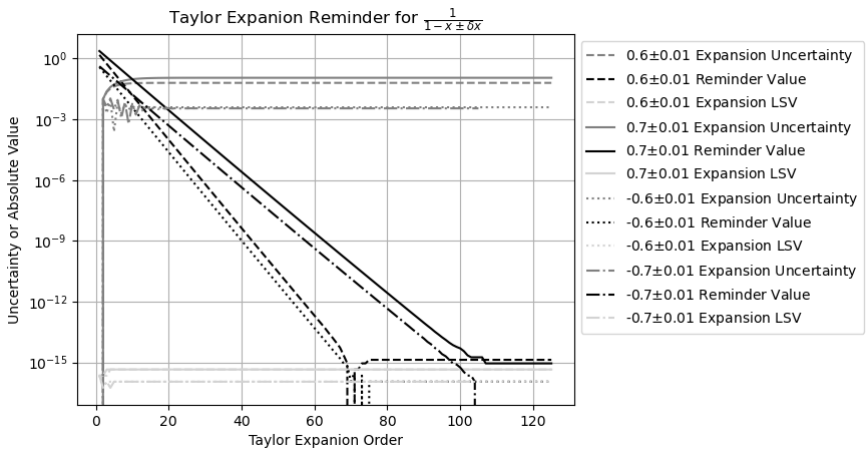

Figure 9: The reminder values and the expansion uncertainties for Taylor expansion of $\frac{1}{1-x\pm\delta x}$, for different Taylor expansion order N as shown in the x-axis, and different input $x \pm \delta x$ as shown in the legend. In the legend, *Reminder Value* is the value of Formula (4.3), *Expansion LSV* is the *LSV* of the value of Formula (4.2), *Expansion Uncertainty* is the uncertainty of Formula (4.2).

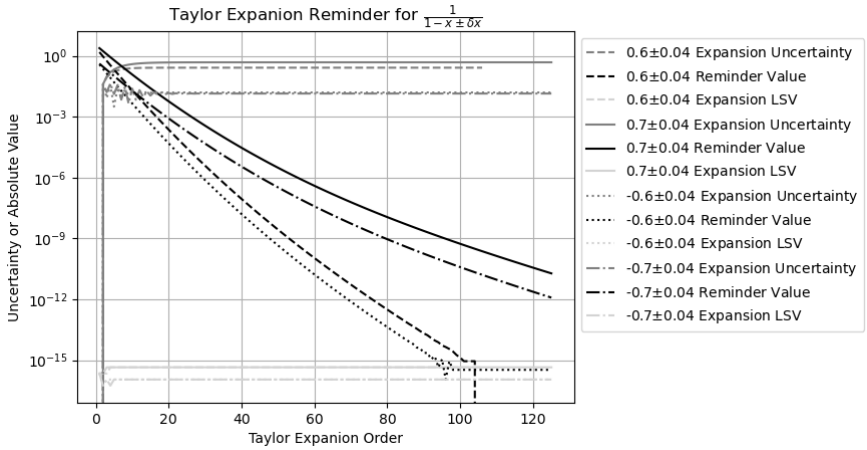

Figure 10: The reminder values and the expansion uncertainties for Taylor expansion of $\frac{1}{1-x\pm\delta x}$, for different Taylor expansion order N as shown in the x-axis, and different input $x \pm \delta x$ as shown in the legend. In the legend, *Reminder Value* is the value of Formula (4.3), *Expansion LSV* is the *LSV* of the value of Formula (4.2), *Expansion Uncertainty* is the uncertainty of Formula (4.2).

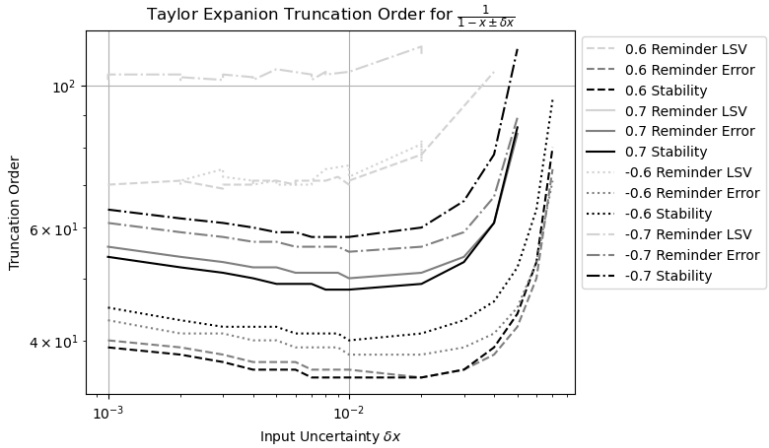

Figure 11: The truncation orders for Taylor expansion of $\frac{1}{1-x\pm\delta x}$, for different input uncertainty δx as shown in the x-axis, and different input value x and truncation rule as shown in the legend. In the legend, *Reminder LSV* means when the value of $r(N)$ is less than the *LSV* of the value of $\frac{1}{1-x\pm\delta x}$, *Reminder Error* means when the absolute value change from $e(N-1)$ to $e(N)$ is less than τ-fold of the uncertainty of $1/(1-x)$, *Stability* means to use the stability criterion to decide maximal N for $e(N)$.

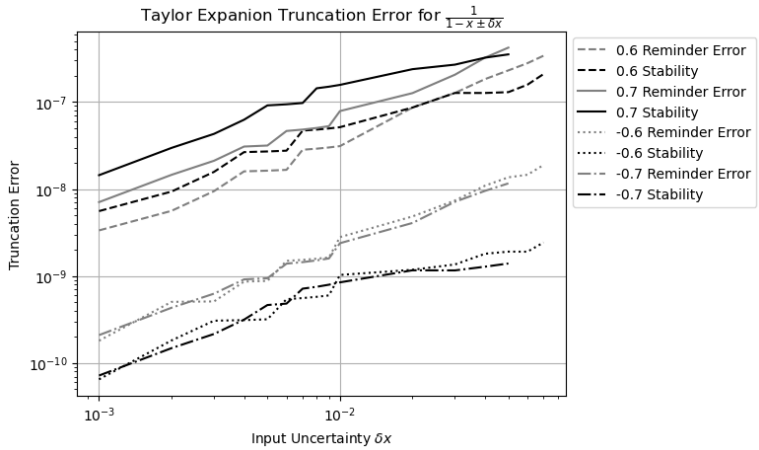

Figure 12: The truncation errors for Taylor expansion of $\frac{1}{1-x\pm\delta x}$, for different input uncertainty δx as shown in the x-axis, and different input value x and truncation rule as shown in the legend. In the legend, *Reminder Error* means when the absolute value change from $e(N-1)$ to $e(N)$ is less than τ-fold of the uncertainty of $1/(1-x\pm\delta x)$, *Stability* means to use the stability criterion to decide maximal N for $e(N)$.

of 70 in Figure 9. The reason for such changes is due to the increase of uncertainty bias in Formula (4.1).

It is questionable to use the expansion truncation standard for precise input to imprecise input. Specifically, it is questionable to always calculate the result value to 10^{-16} precision while the result precision is only about 0.1 for $1/(1+0.6\pm0.04) = 2.526\pm0.26$. Because variance arithmetic allows statistical bounding leakage of $\epsilon = 5.73\times10^{-7}$ according to Formula (2.23), the precision on the result value should be likewise statistically. Formula (4.4) gives the *required value precision* τ, in which $\xi(z)$ is the cumulative density function for Normal distribution [23].

$$2\xi(\tau) = \epsilon = 2 - 2\xi(\frac{1}{2}+\sigma): \quad \tau \simeq \epsilon\frac{\sqrt{2\pi}}{2} = 7.18\times10^{-7}; \tag{4.4}$$

Using τ as the tolerance, the rule for *stability truncation* for truncating an expansion $e(N)$ at the expansion order N is the following:

- The absolute uncertainty change from $e(N-1)$ to $e(N)$ is less than τ-fold of the uncertainty of $e(N)$, and
- The absolute value change from $e(N-1)$ to $e(N)$ is less than either τ-fold of the uncertainty of $e(N)$, or the LSV of the value of $e(N)$.

The stability truncation is free from reminder estimator, and it is based on the stability of $e(N)$ at the expansion order N. The true reminder $r(N)$ at the truncation is renamed as the truncation error.

Figure 11 shows the stability truncation orders for Taylor expansion of $1/(1-x\pm\delta x)$, while Figure 12 shows the corresponding truncation errors. To validate stability truncation in this case, the truncation marked as *Reminder Error* is calculated using the same rule for stability truncation but using the uncertainty of $1/(1-x)$ instead of the uncertainty of $e(N)$. Figure 11 and 12 shows that both truncation method have very similar results, so that truncation without reminder estimator is validated in this case. Figure 12 shows that truncation errors are linearly proportional to input uncertainties, which serves as another validation.

As an comparison, the truncation marked as *Reminder LSV* in Figure 11 is the traditional truncation method, whose truncation orders generally double the corresponding stability truncation orders, showing that stability truncation is more efficient.

Figure 11 shows that for each case, the truncation order has a global minimum, and on each side, either increases slowly with decreasing input uncertainty, or increases fast with increasing input uncertainty. Even when the input uncertainty is comparable to the floating-point rounding error, stability truncation still works by comparing the expansion value change with the LSV of the value after the expansion uncertainty has been stable.

Due to the digital limitation of the 64-bit standard floating-point representation, $\zeta(2n)$ can only be calculated up to $2n = 252$, so that Formula (4.1) can only be calculated for $n < 126$. If after 126 orders of expansion, stability truncation can still not be achieved, the expansion is deemed to be *practically unstable*.

The application of Taylor expansion for uncertainty and uncertainty bias is more than polynomial expansion as described in Formula (4.1). Besides stability truncation and practical stability check, it also has convergence check and reliability check, as described in Section 7. In Figure 11 and Figure 12, $1/(1-0.7\pm\delta x)$ terminates at $\delta x \geq 0.06$ not due to practically unstable, but due to the divergence of $1/(1-0.7\pm\delta x)$ when $\delta x \geq 0.06$. Even after $1/(1-0.7\pm\delta x)$ diverges, Formula (4.1) still gives numerical

results because it does not have convergence check, such as $1/(1 - 0.7 \pm 0.09) = 1.453 \pm 0.197$, which would have given runaway errors in Figure 12. This example shows that the context of a calculation is as import as the result of the calculation. The uncertainty is the most important context of any value, and that is the very reason why variance arithmetic is superior than traditional floating-point calculations for values only.

5 Matrix Inversion

5.1 Uncertainty Propagation in Matrix Determinant

Let vector $[p_1, p_2 \ldots p_n]_n$ denote a permutation of the vector $(1, 2 \ldots n)$ [36]. Let $\$[p_1, p_2 \ldots p_n]_n$ denote the permutation sign of $[p_1, p_2 \ldots p_n]_n$ [36]. Formula (5.1) [36] defines the determinant of a n-by-n square matrix $\mathbf{M}$ with the element $x_{i,j}, i, j = 1 \ldots n$. The sub-determinant [36] $M_{i,j}$ at index (i, j) is formed by deleting the row i and column j of M, whose determinant is given by Formula (5.2) [36]. Formula (5.3) holds for the arbitrary row index i or the arbitrary column index j [36].

$$|\mathbf{M}| \equiv \sum_{[p_1 \ldots p_n]_n} \$[p_1 \ldots p_n]_n \prod_{k=1 \ldots n} x_{k,p_k}; \tag{5.1}$$

$$|\mathbf{M}|_{i,j} \equiv \sum_{[p_1 \ldots p_n]_n}^{p_i = j} \$[p_1 \ldots p_n]_n \prod_{k=1 \ldots n}^{k \neq i} x_{k,p_k}; \tag{5.2}$$

$$|\mathbf{M}| = \sum_{j=1 \ldots n} |\mathbf{M}_{i,j}| x_{i,j} = \sum_{i=1 \ldots n} |\mathbf{M}_{i,j}| x_{i,j}; \tag{5.3}$$

Assuming $p_1, p_2 \in \{1 \ldots n\}$, let $[p_1, p_2]_n$ denote the length-2 unordered permutation which satisfies $p_1 \neq p_2$, and let $< p_1, p_2 >_n$ denote the length-2 ordered permutation which satisfies $p_1 < p_2$, and let $< i_1, i_2 >_n$ be an arbitrary ordered permutation [10]. Formula (5.3) can be applied to $M_{i,j}$, as:

$$|\mathbf{M}_{<i_1,i_2>_n,[j_1,j_2]_n}| \equiv \sum_{[p_1 \ldots p_n]_n}^{p_{i_1}=j_1, p_{i_2}=j_2} \$[p_1 \ldots p_n]_n \prod_{k=1 \ldots n}^{k \notin \{i_1,i_2\}} x_{k,p_k}; \tag{5.4}$$

$$|\mathbf{M}| = \sum_{j_1=1 \ldots n} x_{i_1,j_1} |\mathbf{M}_{i_1,j_1}| = \sum_{j_1=1 \ldots n} \sum_{j_2=1 \ldots n}^{j_2 \neq j_1} |\mathbf{M}_{<i_1,i_2>_n,[j_1,j_2]_n}| x_{i_1,j_1} x_{i_2,j_2}; \tag{5.5}$$

The definition of a sub-determinant can be extended to Formula (5.6), in which $m = 1 \ldots n$. Formula (5.5) can be generalized as Formula (5.7), in which $< i_1 \ldots i_m >_n$ is an arbitrary ordered permutation. The $(n-m)$-by-$(n-m)$ matrix in $|\mathbf{M}_{<i_1 \ldots i_m>_n [j_1 \ldots j_m]_n}|$ is obtained by deleting the rows in $\{i_1 \ldots i_m\}$ and the columns in $\{j_1 \ldots j_m\}$. This leads to Formula (5.8) [11].

$$|\mathbf{M}_{<i_1 \ldots i_m>_n,[j_1 \ldots j_m]_n}| \equiv \sum_{[p_1 \ldots p_n]_n}^{k \in \{i_1 \ldots i_m\}: p_k = j_k} \$[p_1 \ldots p_n]_n \prod_{k=1 \ldots n}^{k \notin \{i_1 \ldots i_m\}} x_{k,p_k}; \tag{5.6}$$

$$|\mathbf{M}| = \sum_{[j_1 \ldots j_m]_n} |\mathbf{M}_{<i_1 \ldots i_m>_n,[j_1 \ldots j_m]_n}| \prod_{k=1}^{m} x_{i_k,j_k}; \tag{5.7}$$

$$\left| |\mathbf{M}_{<i_1 \ldots i_m>_n,[j_1 \ldots j_m]_n}| \right| = ||\mathbf{M}_{<i_1 \ldots i_m>_n,<j_1 \ldots j_m>_n}||; \tag{5.8}$$

Formula (5.9) give the Taylor expansion $|\widetilde{\mathbf{M}}|$ of $|\mathbf{M}|$, which leads to Formula (5.10)

[10] An ordered permutation is a sorted combination.

[11] $||\mathbf{M}||$ in Formula (5.8) means absolute value of determinant.

and (5.11) for mean and variance of matrix determinant, respectively.

$$|\widetilde{\mathbf{M}}| = \sum_{[p_1 \dots p_n]_n} \$[p_1 \dots p_n]_n \prod_{i=1 \dots n} (x_{i,p_i} + \tilde{x}_{i,p_i}) \tag{5.9}$$

$$= \sum_{m=0 \dots n} \sum_{<i_1 \dots i_m>_n} \sum_{[j_1 \dots j_m]_n} \mathbf{M}_{<i_1 \dots i_m>_n, [j_1 \dots j_m]_n} \prod_{i=1 \dots m}^{i \in \{i_1 \dots i_m\}} \tilde{x}_{i,p_i};$$

$$\overline{|\mathbf{M}|} = |\mathbf{M}|; \tag{5.10}$$

$$\delta^2 |\mathbf{M}| = \sum_{m=1}^{n} \sum_{<i_1 \dots i_m>_n} \sum_{[j_1 \dots j_m]_n} |\mathbf{M}_{<i_1 \dots i_m>_n, <j_1 \dots j_m>_n}|^2 \prod_{k=1 \dots n}^{i_k \in \{i_1 \dots i_m\}} (\delta x_{i_k, j_k})^2; \tag{5.11}$$

Formula (5.10) and (5.11) assume that the uncertainties of matrix elements are independent of each other. In contrast, for the special matrix in Formula (5.12), after applying Formula (2.49) and (2.50), Formula (5.13) and (5.14) give the result mean and variance, respectively.

$$|\mathbf{M}| = \begin{vmatrix} x, z, y \\ z, y, x \\ y, x, z \end{vmatrix} = 3xyz - x^3 - y^3 - z^3; \tag{5.12}$$

$$\overline{|\mathbf{M}|} = |\mathbf{M}| - 3x(\delta x)^2 - 3y(\delta y)^2 - 3z(\delta z)^2; \tag{5.13}$$

$$\begin{aligned} \delta^2 |\mathbf{M}| = {} & 3(5x^4 - 12x^2yz + 2xy^3 + 2xz^3 + 3y^2z^2)(\delta x)^2 \\ & + 3(5y^4 - 12y^2xz + 2yx^3 + 2yz^3 + 3x^2z^2)(\delta y)^2 \\ & + 3(5z^4 - 12z^2xy + 2zx^3 + 2zy^3 + 3x^2y^2)(\delta z)^2 \\ & + 9(z^2 + 2xy)(\delta x)^2(\delta y)^2 + 9(y^2 + xz)(\delta x)^2(\delta z)^2 + 9(x^2 + 2yx)(\delta y)^2(\delta z)^2 \\ & + 9(5x^2 - 2yz)(\delta x)^4 + 9(5y^2 - 2xz)(\delta y)^4 + 9(5z^4 - 2xy)(\delta z)^4 \\ & + 15(\delta x)^6 + 15(\delta y)^6 + 15(\delta z)^6; \end{aligned} \tag{5.14}$$

5.2 Adjugate Matrix

The square matrix whose element is $(-1)^{i+j}|\mathbf{M}_{j,i}|$ is defined as the *adjugate matrix* [36] $\mathbf{M}^A$ to the original square matrix $\mathbf{M}$. Let $\mathbf{I}$ be the identical matrix for $\mathbf{M}$. Formula (5.15) and (5.16) show the relation of $\mathbf{M}^A$ and $\mathbf{M}$ [36], which only involve addition, subtraction and multiplication.

$$\mathbf{M} \times \mathbf{M}^A = \mathbf{M}^A \times \mathbf{M} = |\mathbf{M}|\mathbf{I}; \tag{5.15}$$

$$|\mathbf{M}|\mathbf{M}^A = |\mathbf{M}^A|\mathbf{M}; \tag{5.16}$$

To test Formula (5.11):

1. A matrix $\mathbf{M}$ is constructed using random integers uniformly distributed in the range of $[-2^8, +2^8]$, which has a distribution deviation of $2^8/\sqrt{3}$. The integer arithmetic grantees that $|\mathbf{M}|$, $\mathbf{M}^A$, and $|\mathbf{M}^A|$ are precise.
2. Gaussian noises of predefined levels are added to $\mathbf{M}$, to construct a $\widetilde{\mathbf{M}}$ which contains imprecise values. Variance arithmetic is used to calculate $|\widetilde{\mathbf{M}}|$, $\widetilde{\mathbf{M}}^A$, and $|\widetilde{\mathbf{M}}^A|$. For example, to construct a $\widetilde{\mathbf{M}}$ with 10^{-3} input noise precision, the deviation of the Gaussian noise is $10^{-3} \times 2^8/\sqrt{3}$.

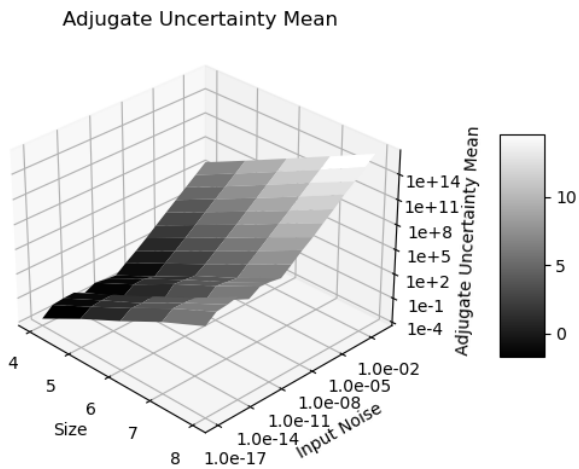

Figure 13: The result uncertainty means vs. input noise precision and matrix size for the difference between $\widetilde{\mathbf{M}}^A$ and $\mathbf{M}^A$. The input noise precision runs from 10^{-17} to 10^{-1}, while the matrix size runs from 4 to 8.

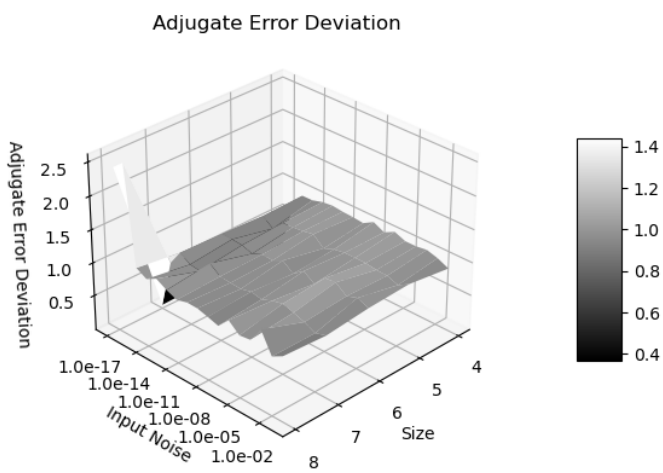

Figure 14: The result error deviations vs. input noise precision and matrix size for the difference between $\widetilde{\mathbf{M}}^A$ and $\mathbf{M}^A$. The input noise precision runs from 10^{-17} to 10^{-1}, while the matrix size runs from 4 to 8.

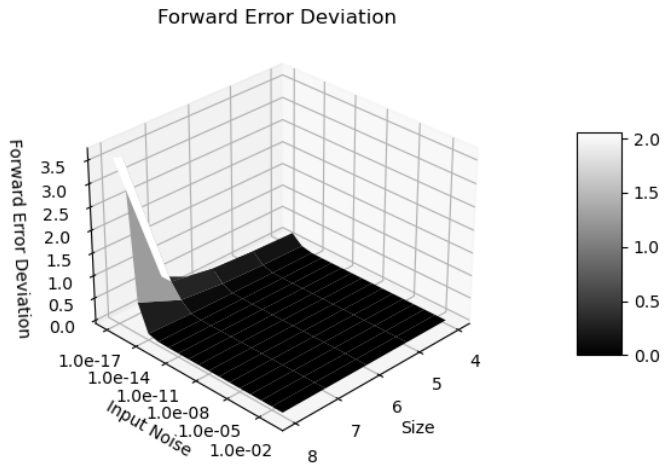

Figure 15: The result error deviations vs. input noise precision and matrix size for the difference of Formula (5.15). The input noise precision runs from 10^{-17} to 10^{-1}, while the matrix size runs from 4 to 8.

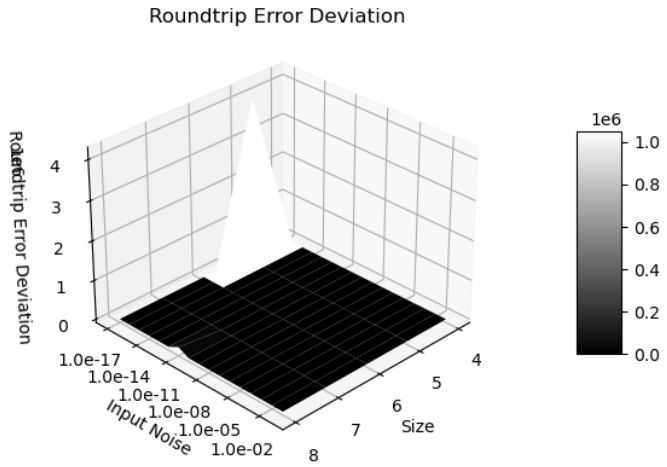

Figure 16: The result error deviations vs. input noise precision and matrix size for the difference of Formula (5.16). The input noise precision runs from 10^{-17} to 10^{-1}, while the matrix size runs from 4 to 8.

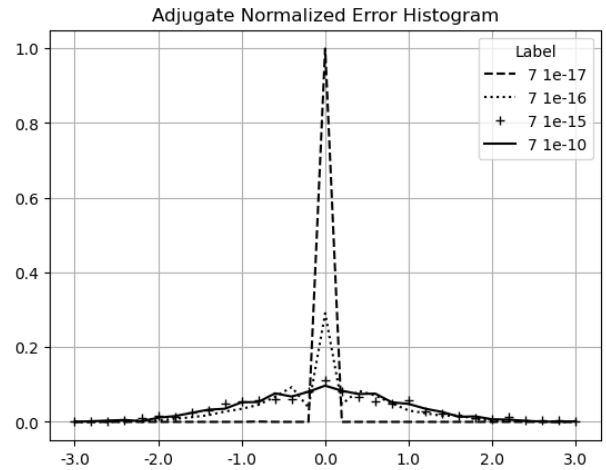

Figure 17: The result histograms for the normalized errors of the adjugate matrix for the same matrix size 7 and different input noise precision as shown in the legend.

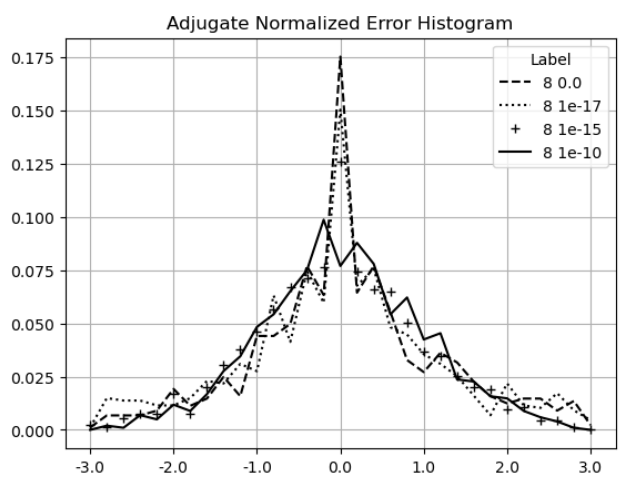

Figure 18: The result histograms for the normalized errors of the adjugate matrix for the same matrix size 8 and different input noise precision as shown in the legend.

Error Deviation	Input Noise Precision			
Matrix Size	0	10^{-17}	10^{-16}	10^{-15}
4	0	0	0.68	1.03
5	0	0.003	0.60	0.96
6	0	0.006	0.80	1.04
7	0	0.083	1.05	1.05
8	4.62	3.91	3.01	1.01

Table 2: The error deviation as a function of matrix size and input noise precision for $\mathbf{M}^A$ in which $\mathbf{M}$ is generated by random integers between $[-2^8, +2^8]$. The measured error deviations change slightly for different runs.

3. The difference between $\widetilde{\mathbf{M}}^A$ and $\mathbf{M}^A$ defines the *Adjugate* Test. Figure 13 shows that the result uncertainties increase exponentially with both the input noises and the matrix size, while Figure 14 shows that the ideal coverage is achieved for the input precision greater than 10^{-16}.
4. Formula (5.15) defines the *Forward* Test. Figure 15 shows that the differences of the two sides of Formula (5.15) is true zero, confirming Formula (5.15) in the context of ideal converge in Figure 14.
5. Formula (5.16) defines the *Roundtrip* Test. Similarly, Figure 16 shows that Formula (5.16) is strictly observed for the idea coverage.

The existence of ideal coverage validates Formula (5.11).

5.3 Floating Point Rounding Errors

The significand of the conventional floating-point representation [7] has 53-bit resolution, which is equivalent to 10^{-16}. Adding noise below this level will not change the value of a conventional floating-point value, so the result uncertainties are caused by the round errors in the floating-point calculations.

Figure 17 shows that the histograms of the normalized errors is Delta distributed for the input uncertainty 10^{-17}, because the adjugate matrix calculation involves about $8 \times 6 = 48$ significand bits for a matrix size 7. Such Delta distribution also holds when the matrix size is less than 7, or when the input noise precision is 0. When the matrix size is 8, $8 \times 7 = 56$ significand bits are needed so rounding occurs. As a result, in Figure 18, the distribution becomes Gaussian with a hint of Delta distribution. Such delta-like distribution persists until the input noise precision reaches 10^{-10}, which is also the transition of the two trends in Figure 13.

When the input noise precision increases from 10^{-17} to 10^{-10}, the distributions become less delta-like and more Gaussian, as shown in both Figure 17 and Figure 18. The non-ideal behaviors in Figure 14, 15, and 16 are all attributed to the rounding errors. Table 2 shows that variance arithmetic achieves proper coverage for the calculation rounding errors for adjugate matrix.

Figure 18 suggests that an histogram of normalized errors is more sensitive to rounding errors than the corresponding error deviation, due to averaging effect of the latter. Histograms of normalized errors can be used as a finger-print for the associated calculation. In case error deviation is not known, a nearly Normal-distributed result histogram may be used to identify ideal coverage.

5.4 Matrix Inversion

A major usage of $|\mathbf{M}|$ and $\mathbf{M}^A$ is to define $\mathbf{M}^{-1}$ in Formula (5.17) which satisfies Formula (5.18). Formula (5.19) gives the Taylor expansion for the element $\mathbf{M}^{-1}_{j,i}$. From it, $\overline{\mathbf{M}^{-1}_{j,i}}$ and $\delta^2\mathbf{M}^{-1}_{j,i}$ can be deduced, in the same way as how Formula (2.17) and (2.18) are deducted from Formula (2.14). Although the analytic expression for $\overline{|\mathbf{M}|}$ and $\delta^2|\mathbf{M}|$ is very complex on paper, the deduction is straight-forward in analytic programming such as using *SymPy* [12].

$$\mathbf{M}^{-1} \equiv \mathbf{M}^A/|\mathbf{M}|; \tag{5.17}$$

$$(\mathbf{M}^{-1})^{-1} = \mathbf{M}; \tag{5.18}$$

$$\begin{aligned}\widetilde{\mathbf{M}^{-1}}_{j,i} &= (-1)^{i+j}\frac{\widetilde{|\mathbf{M}_{i,j}|}}{\widetilde{|\mathbf{M}|}} \qquad (5.19)\\ &= (-1)^{i+j}\left(\sum_{m=0}^{n}\sum_{<i_1\ldots i_m>_n}^{i\in\{i_1\ldots i_m\}}\sum_{[j_1\ldots j_m]_n}^{j_i=j} M_{<i_1\ldots i_m>_n,[j_1\ldots j_m]_n}\prod_{k=1\ldots m}^{k\in\{i_1\ldots i_m\}}\tilde{x}_{k,p_k}\right)\\ &\quad \sum_{h=0}^{\infty}\frac{(-1)^h}{|M|^{h+1}}\left(\sum_{m=1}^{n}\sum_{<i_1\ldots i_m>_n}\sum_{[j_1\ldots j_m]_n} M_{<i_1\ldots i_m>_n,[j_1\ldots j_m]_n}\prod_{k=1\ldots m}^{k\in\{i_1\ldots i_m\}}\tilde{x}_{k,p_k}\right)^h;\end{aligned}$$

Formula (5.19) suggests that $\overline{\mathbf{M}^{-1}} \neq \overline{\mathbf{M}^A}/|\mathbf{M}|$, and $\delta^2\mathbf{M}^{-1}$ is much larger than $\delta^2\mathbf{M}$, which agrees with the common practice to avoid using Formula (5.17) directly [11].

Traditionally, the error response of $\mathbf{M}^{-1}$ is quantified by matrix condition number [36]. In Formula (5.17), $\mathbf{M}^{-1}$ is dominated by $1/|\mathbf{M}|$, suggesting that the precision of $\mathbf{M}^{-1}$ is largely determined by the precision of $|\mathbf{M}|$. Variance arithmetic treats the *LSV* to be imprecise, so it can use Formula (5.14) to calculate the determinant variance for a matrix containing floating-point elements, including a randomly generated matrix, or a Hilbert matrix [36] after it is converted to a floating-point matrix [13]. Figure 19 shows that there is a strong linear correlation between the conditional number and the determinant precision of matrices, so that the determinant precision can replace the condition number, which is more difficult to calculate [14].

In fact, condition number may not be the best tool to describe the error response of a matrix. Formula (5.20) shows an example matrix inversion using variance arithmetic representation. It clearly shows the overall precision dominance by the value of 2 ± 0.161 on the denominator, e.g., 1 ± 0.1 in the adjugate matrix becomes -0.510 ± 0.180 in the inverted matrix, while the direct calculation of $1\pm0.1/-2\pm0.161 = -0.503\pm0.065$ has 3-fold better result precision. In Formula (5.20), even though the input matrix has element precision ranging from 10^{-1} to $2.5\ 10^{-5}$, the inverted matrix has element precision all near 0.2 except the worst one at 0.35, showing how dramatically the element precision gets worse in matrix inversion. Formula (5.19) suggests that the

[12] The result of using SymPy for matrix inversion is demonstrated as part of an open source project at https://github.com/Chengpu0707/VarianceArithmetic.

[13] Although Hilbert matrix is defined using fractional number, its condition number is also calculated using floating-point arithmetic in math libraries such as *SciPy*.

[14] There are currently different definitions of matrix condition number, each based on a different definition of matrix norm.

precision of inverted matrix deteriorates exponentially with matrix size.

$$\begin{aligned}\begin{pmatrix}1\pm10^{-1}, & 2\pm10^{-2}\\ 3\pm10^{-3}, & 4\pm10^{-4}\end{pmatrix}^{-1} &= \frac{\begin{pmatrix}4\pm10^{-4}, & -2\pm10^{-2}\\ -3\pm10^{-3}, & 1\pm10^{-1}\end{pmatrix}}{-2\pm0.161}\\ &\simeq \begin{pmatrix}-2.000\pm0.401, & 1.000\pm0.201\\ 1.500\pm0.301, & -0.510\pm0.180\end{pmatrix};\end{aligned} \tag{5.20}$$

5.5 First Order Approximation

Formula (5.21) shows the first order approximation of $|\widetilde{\mathbf{M}}|$ leads to the first order approximation of $\delta^2|\mathbf{M}|$. It essentially states that when the input precision is much less than 1, the determinant $|\mathbf{M}|$ of an imprecise matrix $\mathbf{M}$ can be calculated in variance arithmetic using Formula (5.1) directly.

$$|\widetilde{\mathbf{M}}| \simeq \sum_{[p_1\dots p_n]_n} \$[p_1\dots p_n]_n \prod_{k=1\dots n}(x_{k,p_k}+\tilde{x}_{k,p_k}); \quad\Rightarrow\quad \delta^2|\mathbf{M}| \simeq \sum_i^n\sum_j^n M_{i,j}(\delta x_{i,j})^2; \tag{5.21}$$

Figure 20 contains the result of applying Formula (5.21). It is very similar to Figure 14, validating Formula (5.21).

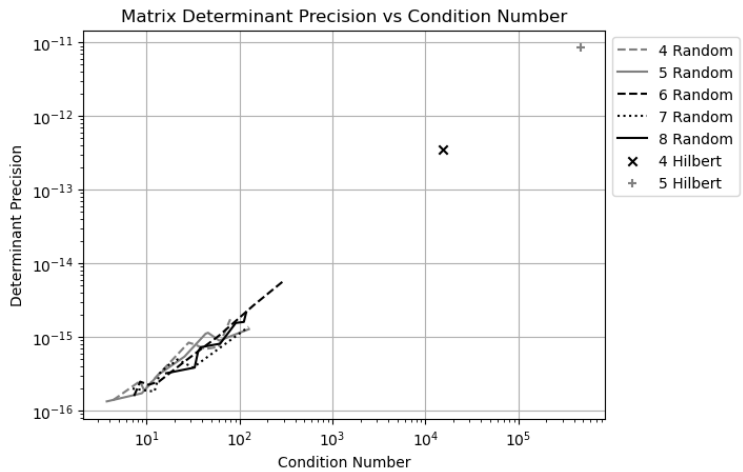

Figure 19: The linear correlation between the precision of a matrix determinant to its condition number. The x-axis shows the condition number, while the y axis shows the precision of the determinant. The legend shows the size of the matrix, as well as the type of the matrix as *Random* for randomly generated matrix, and *Hilbert* as the Hilbert matrix.

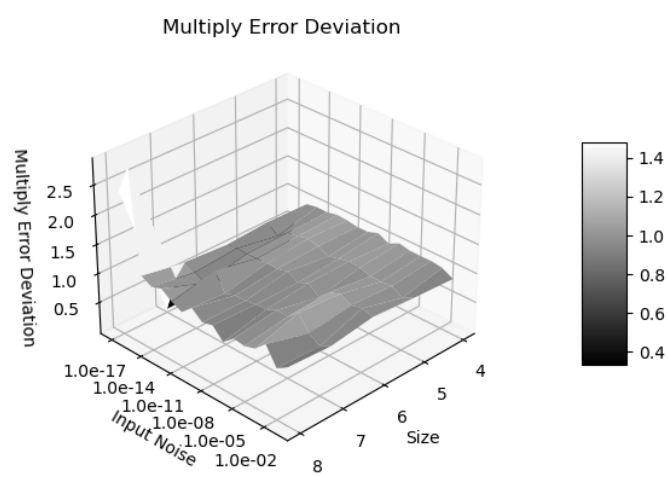

Figure 20: The error deviation of the first approximation calculation of $|\mathbf{M}|$ vs. input noise precision and matrix size.

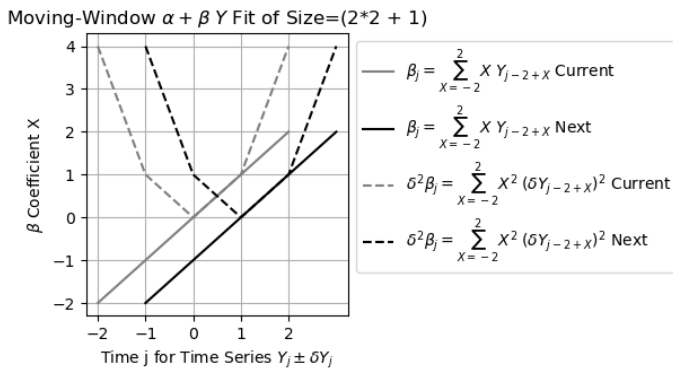

Figure 21: Coefficients of X in Formula (6.4) and (6.8) at a current and the next position in a time series of the $\alpha + \beta\ Y$ fit. It shows that the β_{j+1} in Formula (6.4) can be calculated progressively from β_j, as Formula (6.6) and (6.5) in order. It also shows that the $\delta^2\beta_{j+1}$ in Formula (6.8) cannot be calculated easily from $\delta^2\beta_j$.

6 Moving-Window Linear Regression

6.1 Moving-Window Linear Regression Algorithm [1]

Formula (6.1) and (6.2) give the result of the least-square line-fit of $Y = \alpha + \beta X$ between two set of data Y_j and X_j, in which j is an integer index to identify (X, Y) pairs in the sets [11].

$$\alpha = \frac{\sum_j Y_j}{\sum_j 1}; \tag{6.1}$$

$$\beta = \frac{\sum_j X_j Y_j\ \sum_j 1 - \sum_j X_j\ \sum_j Y_j}{\sum_j X_j X_j\ \sum_j 1 - \sum_j X_j\ \sum_j X_j}; \tag{6.2}$$

In many applications, data set Y_j is an input data stream collected with fixed rate in time, such as a data stream collected by an ADC (Analogue-to-Digital Converter) [37]. Y_j is called a time-series input, in which j indicates time. A moving window algorithm [11] is performed in a small time-window around each j. For each window of calculation, X_j can be chosen to be integers in the range of $[-H, +H]$ in which H is an integer constant specifying window's half width so that $\sum_j X_j = 0$, to reduce Formula (6.1) and (6.2) into Formula (6.3) and (6.4), respectively [1]:

$$\alpha_j = \alpha\ 2H = \sum_{X=-H+1}^{H} Y_{j-H+X}; \tag{6.3}$$

$$\beta_j = \beta\ \frac{H(H+1)(2H+1)}{3} = \sum_{X=-H}^{H} X Y_{j-H+X}; \tag{6.4}$$

According to Figure 21, in which H takes an example value of 2, the calculation of (α_j, β_j) can be obtained from the previous values of $(\alpha_{j-1}, \beta_{j-1})$, to reduce the

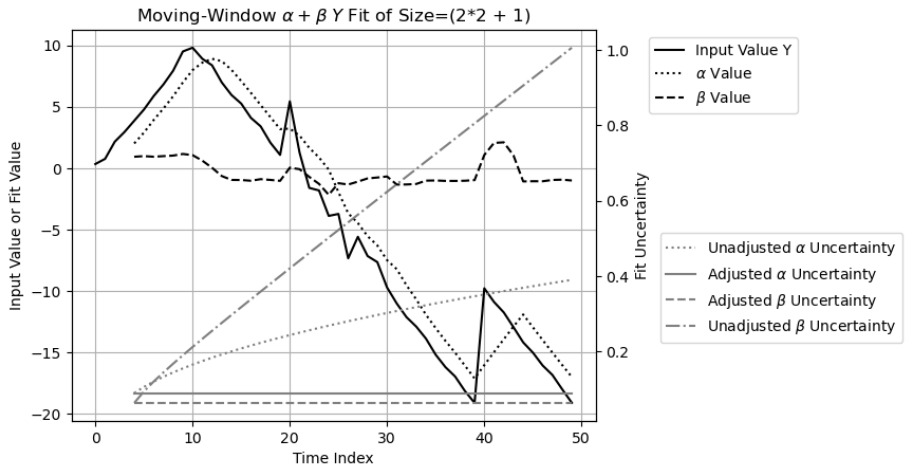

Figure 22: The result of fitting $\alpha + \beta\ Y$ to a time-series input Y within a moving window of size $2 * 2 + 1$. The x-axis marks the time index. The y-axis on the left is for the value of Y, α, and β, while the y-axis on the right is for the uncertainty of α and β. The uncertainty for Y is a constant of 0.2. In the legend, *Unadjusted* means the result of applying Formula (6.5) and (6.6) directly using variance arithmetic, while *Adjusted* means using Formula (6.5) and (6.6) for α and β values but using Formula (6.7) and (6.8) for α and β variances.

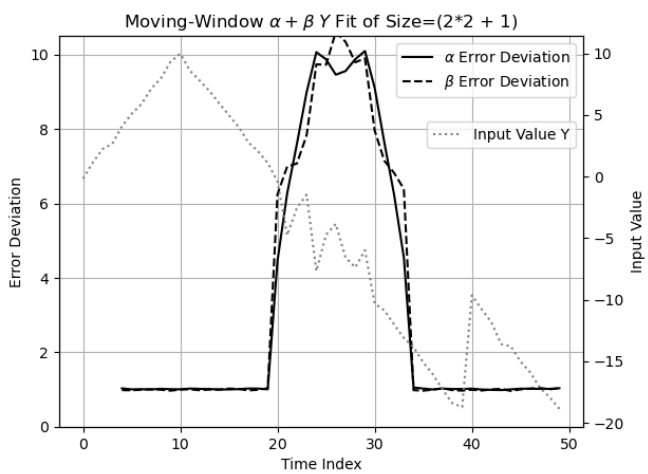

Figure 23: The error deviations of the $\alpha + \beta\ Y$ fit vs time index. The x-axis marks the time index. The y-axis on the left is for error deviation. An input time-series signal Y is also provided for reference, whose value is marked by the y-axis on the right.

calculation of Formula (6.3) and (6.4) into a progressive moving-window calculation of Formula (6.5) and (6.6), respectively:

$$\beta_j = \beta_{j-1} - \alpha_{j-1} + H\left(Y_{j-2H-1} + Y_j\right); \tag{6.5}$$

$$\alpha_j = \alpha_{j-1} - Y_{j-2H-1} + Y_j; \tag{6.6}$$

6.2 Variance Adjustment

When the time series contains uncertainty, the direct application of Formula (6.5) and (6.6) will result in loss of precision, because both formulas use each input multiple times and accumulate the variance of the input once at each usage. Thus, the values for α_j and β_j should be calculated progressively using Formula (6.6) and (6.5), while the variance should be calculated using Formula (6.7) and (6.8), respectively. Formula (6.8) is no longer progressive.

$$\delta^2\alpha_j = \sum_{X=-H+1}^{H} \left(\delta Y_{j-H+X}\right)^2 = \delta^2\alpha_{j-1} + \left(\delta Y_j\right)^2 - \left(\delta Y_{j-2H}\right)^2; \tag{6.7}$$

$$\delta^2\beta_j = \sum_{X=-H+1}^{H} X^2\left(\delta Y_{j-H+X}\right)^2; \tag{6.8}$$

Figure 22 shows that the input signal Y_j is composed of:

1. An increasing slope for $j = 0 \ldots 9$.
2. A decreasing slope for $j = 1 \ldots 39$.
3. A sudden jump with an intensity of $+10$ at $j = 40$
4. A decreasing slope for $j = 41 \ldots 49$.

For each increase of j, the increasing rate and the decreasing rate are $+1$ and -1, respectively.

The specified input uncertainty is always 0.2. Normal noises of 0.2 deviation are added to the slopes, except Normal noises of 2 deviation are added to the slope for $j = 10 \ldots 19$, during which the actual noises is 10-fold of the specified value.

Figure 22 also shows the result of moving window fitting of $\alpha + \beta\, Y$ vs time index j. The result values of α and β behave correctly with the expected delay in j. When Formula (6.3) and (6.4) are used both for values and uncertainties for α and β, the result uncertainties of α and β both increase exponentially with the index time j. When Formula (6.3) and (6.4) are used only for values, while Formula (6.7) and (6.8) are used for variance for α and β, the result uncertainty of α is $\sqrt{\frac{1}{2H+1}}\delta Y$, and the result uncertainty of β is $\sqrt{\frac{3}{H(H+1)(2H+1)}}\delta Y$, both of which are less than the input uncertainty δY, due to the averaging effect of the moving window.

6.3 Unspecified Input Error

To obtain the error deviation of α and β, the fitting is done on many time-series data, each with independent generated noises. Figure 23 shows the corresponding error deviation vs index time j, which are 1 except near 10 for $j = 10 \ldots 19$ when the actual noise is 10-fold of the specified one. It shows that a larger than 1 error deviation may probably indicate unspecified additional input errors other than rounding errors, such as the numerical errors from math libraries.

7 Convergence and Statistical Bounding

7.1 Convergence Criteria

Variance arithmetic rejects a calculation result for the following numerical reasons:

- For Formula (2.18) to converge, the contribution from expansion order $2n$ should decrease monotonically with increasing $2n$ when $2n$ is sufficiently large [15]. Otherwise, the result is invalidated as *not monotonic*.
- An uncertainty only needs to be correct on order of magnitude, such as with a precision of $1/\sigma = 0.2$ or finer. If the precision of the uncertainty becomes worse than $1/\sigma$, the result is invalidated as *not reliable*.
- Due to digital limitation of conventional floating-point representation, $\zeta(2n)$ can only be calculated to $2n \leq 252$. If at the limit, the expansion is still not stable according to stability truncation criterion, the result is invalidated as *practical unstable*.

In reality, "practical unstable" happens very rarely if any.

7.2 Exponential

Because $2n!$ is much larger than $2n!!$, Formula (2.26) for e^x always converges. However, due to digital limitation, $e^{x\pm\delta x}$ is limited before the result uncertainty becomes either 0 wrongly or ∞, such as $e^{-392\pm0.1} = 1.005\ e^{-392} \pm 0$ or $e^{196\pm0.1} = 1.005\ e^{196} \pm \infty$.

7.3 Logarithm

Figure 24 shows that Formula (2.30) for $\ln(x)$ converges when $P(x) < 1/\sigma$, which is also the theoretical upper bound for $P(x)$ because $2n$ can be regarded as a constant when compared with $\zeta(2n)$ in Formula (2.30). Variance arithmetic rejects the distributional pole of $\ln(0)$ inside the range of $[x - \delta x, x + \delta x]$ statistically due to the divergence of Formula (2.30) mathematically, with $\zeta(2n)$ as the connection of these two worlds.

7.4 Power

If c is 0 or a natural number, Formula (2.38) always converges because it only has limited terms, such as Formula (2.41) for $c = 2$. Even though the normalized value errors for $(0 + \delta x)^2$ is χ^2 distributed, its error deviation is 1.0015 ± 0.0059. This shows that the variance arithmetic allows normalized error distributions to deviate from Normal to a large degree, as what Formula (2.22) has suggested.

If c is not a natural number, at $x = 0$, x^c has either a distributional pole when $c < 1$ or a distributional zero when $c > 1$. Figure 25 shows that Formula (2.38) for $(1 \pm \delta x)^c$ also rejects 0^c, but such rejection depends on details of Formula (2.38). Figure 25 plot the upper bound of δx vs c, as well as the corresponding uncertainty bias and result uncertainty, omitting the the cases when c is 0 or a natural number.

- When $c < 1$, according to Formula (2.43), (2.45), and (2.47), the coefficient for $P(x)^{2n}$ increases quickly with both c and $2n$, so that the upper bound for δx has to decrease with c, which is confirmed by Figure 25.

[15] The lower limit for $2n$ is set empirically as 20.

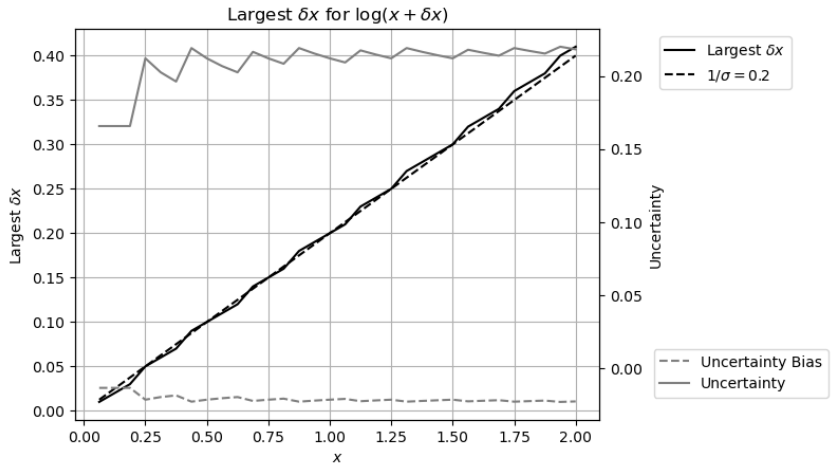

Figure 24: The largest measured δx for $\ln x \pm \delta x$ for different x as shown by the x-axis, using the y-axis on the left. Also shown are the corresponding uncertainty bias and uncertainty for different x as shown by the x-axis, using the y-axis on the right.

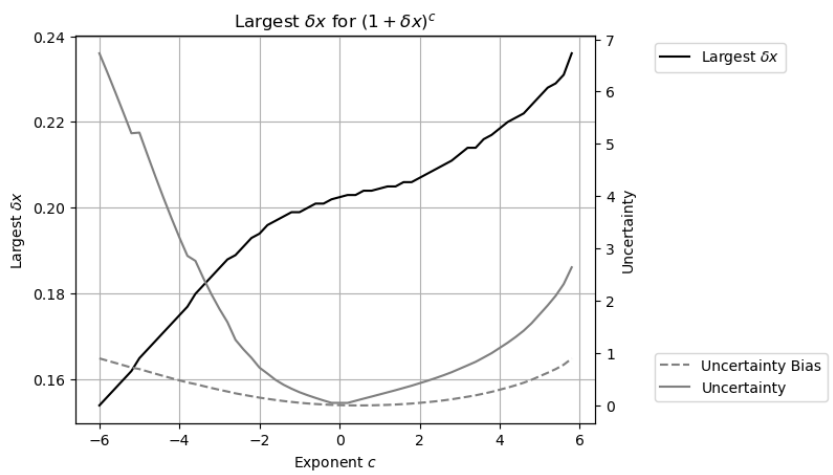

Figure 25: The largest measured δx for $\ln(x \pm \delta x)$ for different x as shown by the x-axis, using the y-axis on the left. Also shown are the corresponding uncertainty bias and uncertainty for different x as shown by the x-axis, using the y-axis on the right.

- When $c > 1$, Figure 25 shows that both result uncertainty and measure upper bound for input uncertainty δx increase with increasing c, but slower than the trend when $c < 1$. This can be explained by the alternative $+$ and $-$ sign pattern in the Taylor expansion, to reduce higher order terms in the expansion of result variance, such as the zero $P(x)^6$ expansion term for $\delta^2\sqrt{x^5}$.

Figure 26 shows that the value converges faster than the variance, due to the difference between Formula (2.17) and (2.18). It shows the individual variance term of $1/(1 \pm \delta x)$ for the different Taylor expansion order $2n$, and δx, in which $1/(1 \pm 0.201)$ clearly diverges, while $1/(1 \pm 0.200)$ marginally diverges, so that variance arithmetic invalidate both as not monotonic because both do not satisfy the necessary condition for convergence. Due to the limitation of maximal expansion order of $2n = 252$, such determination of convergence is only practical, which may be different from theoretical convergence threshold.

7.5 Sine

Because $2n!$ is much larger than $2n!!$, Formula (2.34) for $sin(x)$ always converges.

Figure 27 shows the histogram of normalized errors for $\sin(x \pm 0.001)$ when Gaussian noises are added. The distribution is Normal at $x = 0, \pi/4$, but it is a folded Normal at $x = \pi/2$ when $\sin(x)$ has a distributional zero, and the 2nd derivative of $\sin(x)$ folds all added noises to one side only. Because the 2nd derivative of $\sin(-\pi/2)$ has the opposite sign to that of $\sin(\pi/2)$, the 2nd derivative of $\sin(-\pi/2)$ folds the Normal distribution to the opposite side only.

In Figure 28, the noise δx is increased from 0.001 for $\sin(\pi/4 \pm \delta x)$:

- When $\delta x = 0.001$, the distribution is Normal.
- When $\delta x = 0.2$, the distribution at $x = \pi/4$ starts to be affected by the closest distributional zero at $x = \pi/2$, so that the distribution starts to deviate from Normal distribution.
- When $\delta x = 0.5$, the distribution at $x = \pi/4$ is affected strongly by the closest distributional zero at $x = \pi/2$, so that the distribution starts to resemble of the distribution at $x = \pi/2$ in Figure 27.
- When $\delta x = 1.41$, the distribution at $x = \pi/4$ starts to be affected by the second closest distributional zero at $x = -\pi/2$, which result in a small peak resembling the distribution at $x = -\pi/2$ in Figure 27.
- When $\delta x = 1.42$, the result is invalidated as not reliable, which is a pleasant side effect of the convergence criteria of variance arithmetic. In a periodic function $\sin(x + \delta x)$, δx conceptually should not be larger than period 2π. Variance arithmetic just provides a sensible upper bound for δx in this case.

In some sense, a distribution is like sticky fluid, to be bounced by two enclosing distributional zero walls [16] of $\sin(x)$. Figure 28 shows the distributions for $\sin(\pi/4 + \delta x)$ until reaching the maximal holding capacity of the enclosing distributional zeros. Figure 29 shows how the distribution lava for $\sin(0 + \delta x)$ flows fill evenly between two distributional zero walls with increasing δx. Figure 30 shows how a distributional zero wall of $\sin(\pi/2 + \delta x)$ itself is broken with increasing δx by its enclosing distributional zero wall. For an analytic function with multiple distributional zeros, the monotonic

[16] The relation between a distribution and its enclosing distributional zeros resembles the relation between quantum waves and quantum walls.

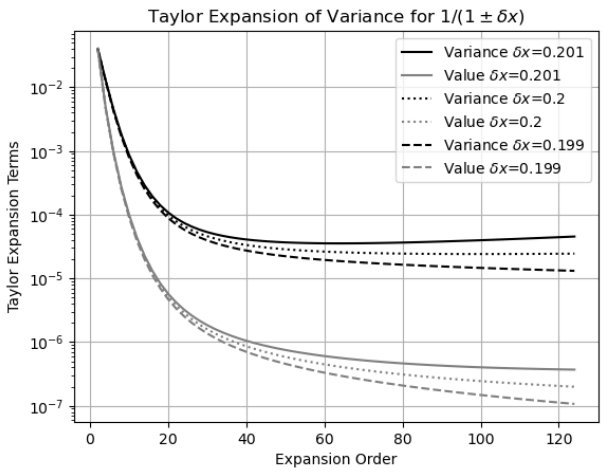

Figure 26: The individual variance term of $1/(1 \pm \delta x)$ vs the different Taylor expansion order $2n$, for different δx as shown in the legend. The x-axis is Taylor expansion order $2n$ Formula (2.18). The y-axis is the value and variance contribution to $1/(1 \pm \delta x)$ at each Taylor expansion order $2n$ according to Formula (2.37), and (2.38), respectively.

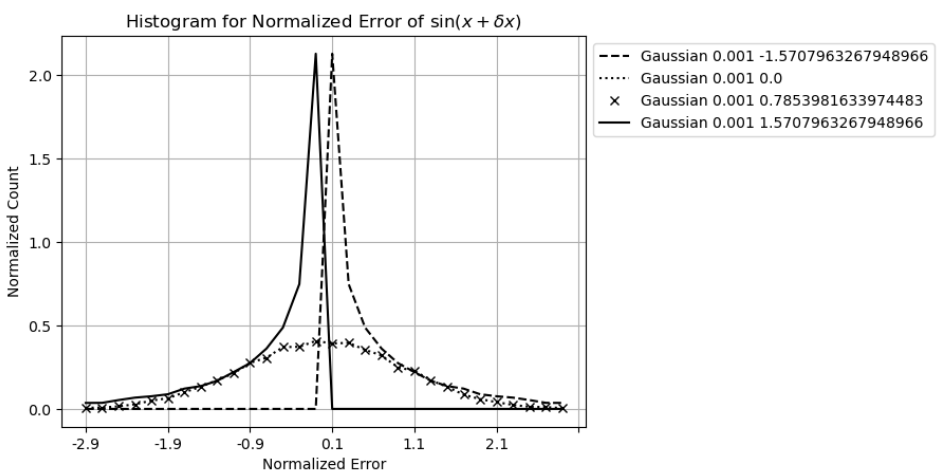

Figure 27: The histogram of the error deviation for $\sin(x \pm \delta x)$, for $x = 0, +\pi/2, +\pi/4$ as shown in the legend. The x-axis is for normalized value error. The y-axis is for normalized count. Each color represents a different x, while each line pattern represents a different δx. Gaussian noises are used to produce the input noise $\delta x = 0.001$.

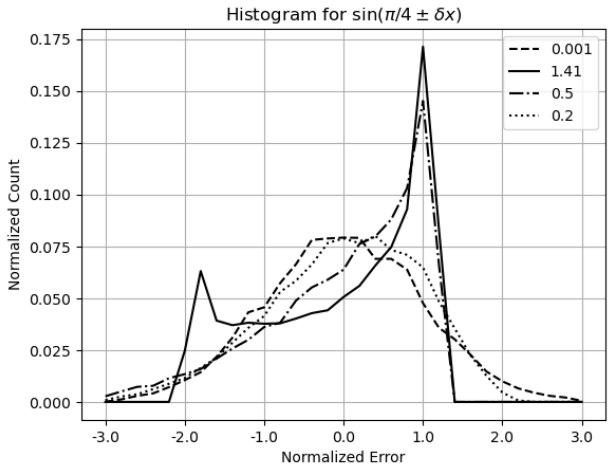

Figure 28: The histogram for $\sin(\pi/4 \pm \delta x)$ for different δx as shown in the legend. The x-axis is for normalized value error. The y-axis is for normalized count.

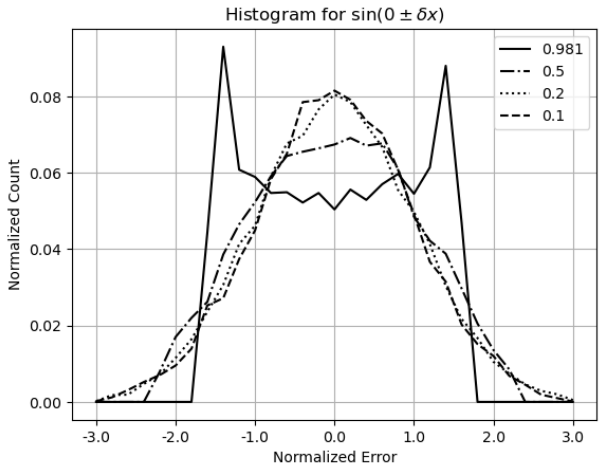

Figure 29: The histogram for $\sin(0 \pm \delta x)$ for different δx as shown in the legend. The x-axis is for normalized value error. The y-axis is for normalized count.

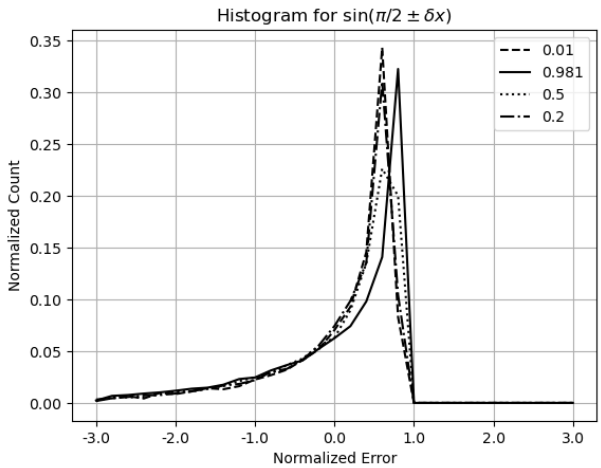

Figure 30: The histogram for $\sin(\pi/2 \pm \delta x)$ for different δx as shown in the legend. The x-axis is for normalized value error. The y-axis is for normalized count.

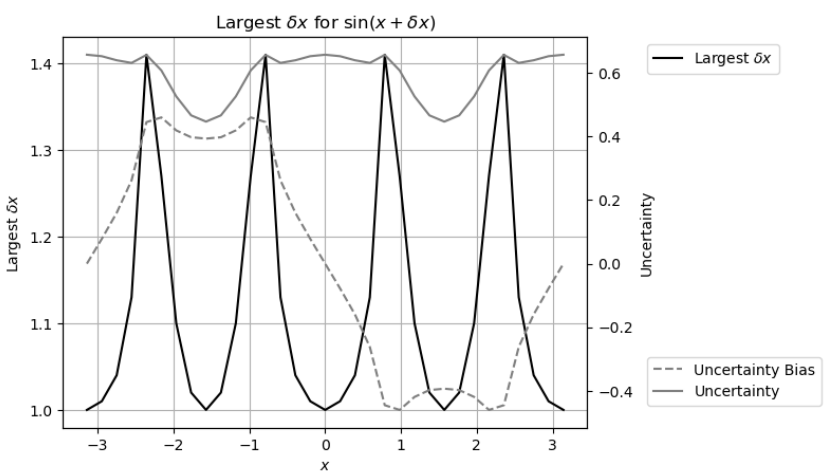

Figure 31: The largest measured δx for $\sin(x \pm \delta x)$ for different x as shown by the x-axis, using the y-axis on the left. Also shown are the corresponding uncertainty bias and uncertainty for different x as shown by the x-axis, using the y-axis on the right.

criterion naturally limits the input uncertainty to the distance between two enclosing distributional zeros.

Figure 31 shows the measured upper bound of δx in $\sin(x + \delta x)$ for $-\pi \leq x \leq \pi$, as well as the corresponding uncertainty and uncertainty bias. The measured upper bound is also periodic with π, with the tightest bounds of δx at the distributional zeros $x = n\pi + \pi/2$. The mechanism for the upper bounding of δx in $\sin(x + \delta x)$ needs further understanding to quantify the result shown in Figure 31.

8 Math Library Functions

Formula (2.26), (2.30), (2.34), and (2.38) are tested by the corresponding math library functions *exp*, *log*, *sin*, and *pow*, respectively.

At each point x for an input uncertainty δx, the result uncertainty is calculated by variance arithmetic. The corresponding error deviation is obtained by sampling:

1. Take 10000 samples from either Gaussian noise or uniform distribution with δx as the deviation, and construct $\tilde{x}$ which is x plus the sampled noise.
2. For each $\tilde{x}$, use the corresponding library function to calculate the value error as the difference between using $\tilde{x}$ and using x as the input.
3. The value error is divided by the result uncertainty, as the normalized error.
4. The standard deviation of the 10000 normalized errors is the error deviation.
5. In addition, for each of these tests, all value errors follow the same underlying distribution. The deviation of the value errors is defined as the *value deviation*, which the uncertainty should match.

8.1 Exponential

Figure 32 shows that the calculated uncertainties using Formula (2.26) agree very well with the measured value deviations for $e^{x+\delta x}$. As a result, all error deviations are very close to 1, as 1.0021 ± 0.0095, even though both the uncertainties and the value error deviations increase exponentially with x and δx.

8.2 Logarithm

$\ln(x)$ has a distributional zero at $x = 0$ so that all $log(x \pm \delta x)$ are rejected when $1/5 < P(x)$. Figure 33 shows that the calculated uncertainties using Formula (2.30) agree very well with the measured value deviations for $\ln(x + \delta x)$, and the result error deviations are very close to 1, as 0.985 ± 0.091.

8.3 Power

Figure 34 shows that the calculated uncertainties of $(1 \pm \delta x)^c$ using Formula (2.38) agree very well with the measured value deviations for $(1 + \delta x)^c$, with the error deviations close to 1, as 0.975 ± 0.135.

8.4 Sine

Figure 35 shows that the calculated uncertainties using Formula (2.34) agree very well with the measured value deviations for $\sin(x + \delta x)$. It also shows that $\delta^2 \sin(x)$ has the same periodicity as $\sin(x)$:

- When $x = 0$, $\sin(x) \simeq x$, so that $\delta^2 \sin(x) \simeq (\delta x)^2$.
- When $x = \pi/2$, $\sin(x) \simeq 1$, so that $\delta^2 \sin(x) \simeq 0$.

$\sin(x)$ has distributional zeros at $x = \pm\pi/2$. Figure 36 shows that the error deviation for $\sin(x + \delta x)$ is 1 except when $x = \pm\pi/2$ and $\delta x < 10^{-8}$, which agrees with the expected Delta distribution at $x = \pm\pi/2$. In other places, the error deviations are very close to 1, as 0.999 ± 0.009.

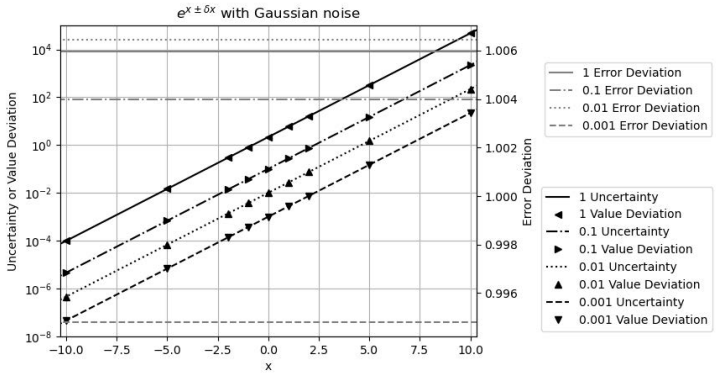

Figure 32: The calculated uncertainties vs the measured value deviations as well as the measured error deviations for $e^{x \pm \delta x}$, for different x as shown by the x-axis, and different δx as shown in the legend. The uncertainties and the value deviations are drawn using the logarithmic y scales on the left side, while the error deviations are drawn using the linear y scales on the right side. Each color represents a δx. Gaussian noises are used to produce the input noise δx.

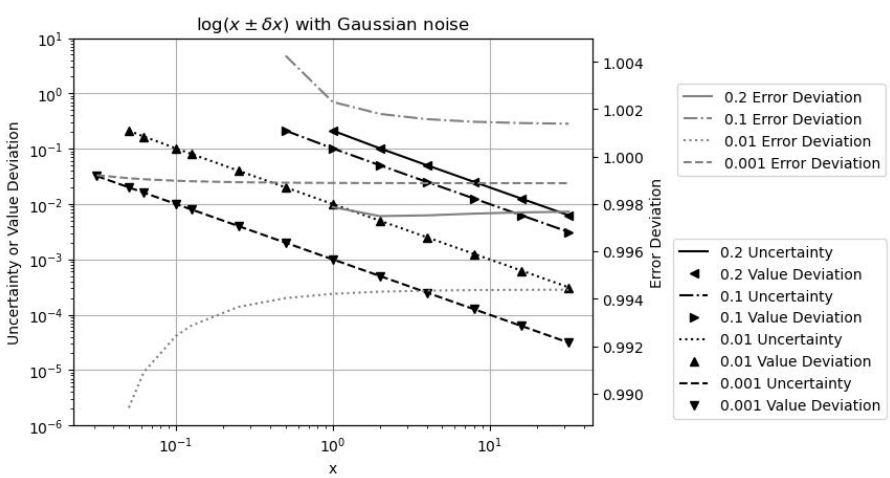

Figure 33: The calculated uncertainties vs the measured value deviations as well as the measured error deviations for $\ln(x \pm \delta x)$, for different x as shown by the x-axis, and different δx as shown in the legend. The uncertainties and the value deviations are drawn using the logarithmic y scales on the left side, while the error deviations are drawn using the linear y scales on the right side. Each color represents a δx. Gaussian noises are used to produce the input noise δx.

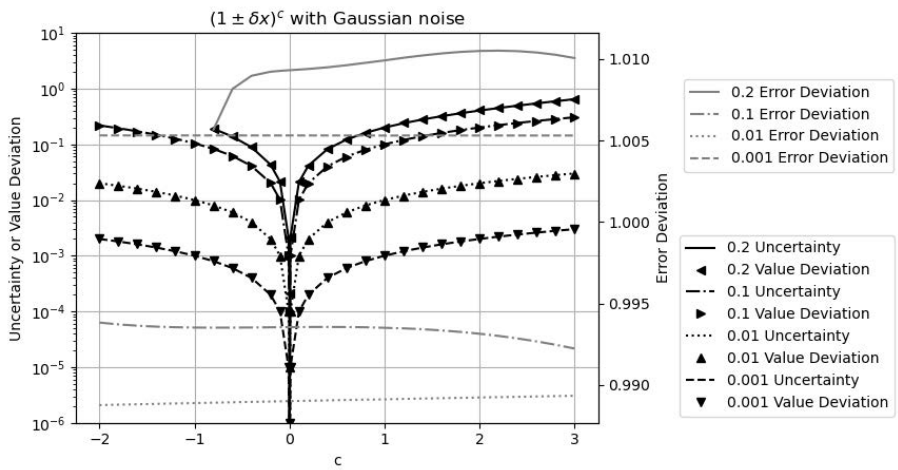

Figure 34: The calculated uncertainties vs the measured value deviations as well as the measured error deviations for $(1 \pm \delta x)^c$, for different c as shown by the x-axis, and for different δx as shown in the legend. The uncertainties and the value deviations are drawn using the logarithmic y scales on the left side, while the error deviations are drawn using the linear y scales on the right side. Each color represents a δx.

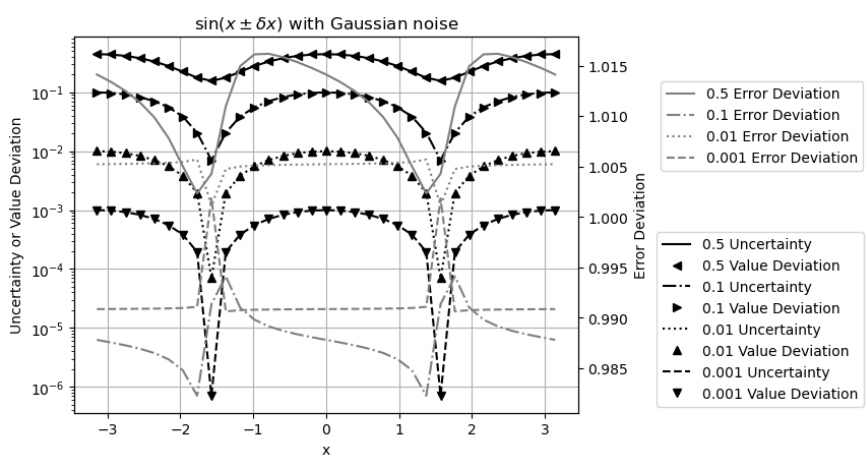

Figure 35: The calculated uncertainties vs the measured value deviations as well as the measured error deviations for $\sin(x \pm \delta x)$, for different x as shown by the x-axis, and different δx as shown in the legend. The uncertainties and the value deviations are drawn using the logarithmic y scales on the left side, while the error deviations are drawn using the linear y scales on the right side. Each color represents a δx. Gaussian noises are used to produce the input noise δx.

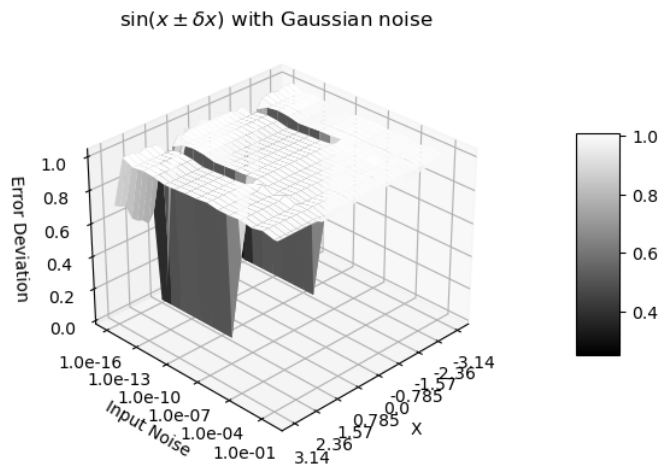

Figure 36: The error deviation for $\sin(x \pm \delta x)$ vs. x and δx. The x-axis is x between $-\pi$ and $+\pi$. The y-axis is δx between -10^{-16} and 1. The z-axis is the error deviation. Gaussian noises are used to produce the input noise δx.

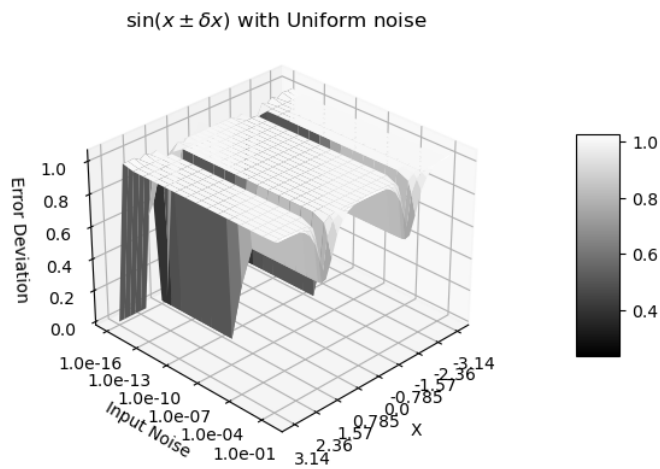

Figure 37: The error deviation for $\sin(x \pm \delta x)$ vs. x and δx. The x-axis is x between $-\pi$ and $+\pi$. The y-axis is δx between -10^{-16} and 1. The z-axis is the error deviation. Uniform noises are used to produce the input noise δx.

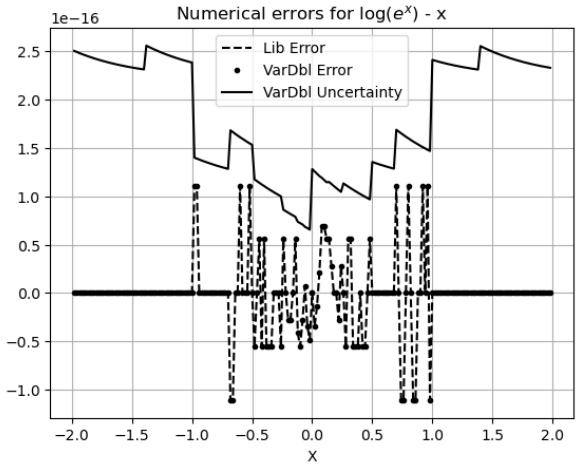

Figure 38: The values and uncertainties of $\ln(e^x) - x$ vs x, as *VarDbl Error* and *VarDbl Uncertainty* in the legend. The result of the same calculation using conventional floating-point library functions is shown as *Lib Error* in the legend.

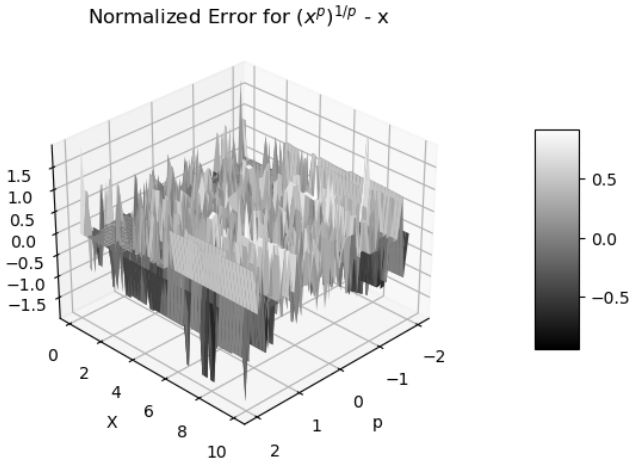

Figure 39: The normalized errors of $(x^p)^{\frac{1}{p}} - x$ vs x and p.

Figure 36 and Figure 37 both show the result uncertainty of $\sin(x + \delta x)$ vs the input x and the input uncertainty δx. In Figure 36, Gaussian noises are used, while in Figure 37, the input noises are uniformly distributed. The difference of Figure 36 and 37 shows that the uniform noise or white noise [17] is more local, while Gaussian noise has strong tail effect, so that the error deviation is much closer to 1 for $\sin(\pm\pi/2+\delta x)$ when $\delta x > 10^{-12}$. In both cases, there is a clear transition at $\delta x = 10^{-12}$, which is caused by numerical errors in library $\sin(x)$, as demonstrated in Section 9. For consistency, only Gaussian noises will be used for other analysis by default.

8.5 Numerical Errors for Library Functions

The combined numerical error of the library function e^x and $\ln(x)$ is calculated as $\ln(e^x) - x$. Figure 38 shows that using either variance arithmetic or the conventional floating-point library functions results in the same value errors. The uncertainties of variance arithmetic bound the value errors effectively, resulting in an error deviation about 0.409 when $|x| \leq 1$. It looks suspicious when $|x| > 1$ the value error is always 0 in both calculations.

The numerical error of the library function x^p is calculated as $(x^p)^{1/p} - x$. Figure 39 shows that the normalized errors are not specific to either x or p, resulting in an error deviation 0.548 ± 0.8.

The numerical errors of the library functions $\sin(x)$, $\cos(x)$, and $\tan(x)$ will be studied in detail later in Section 9.

8.6 Summary

Formula (2.26), (2.30), (2.34), and (2.38) give effective uncertainties for the corresponding library functions. With added noise larger than 10^{-15}, ideal coverage is achievable unless near a distributional pole where the deviation is 0. In non-ideal coverage cases, the error deviation is about 0.5 so that proper coverage is achieved.

[17] White noise is sampled from uniform distributed noise.

9 Fast Fourier Transformation

9.1 Unfaithful Frequency Response of Discrete Fourier Transformation

Each testing algorithm needs to come under careful scrutiny. One important issue is whether the digital implementation of the algorithm is faithful to the original analytic algorithm. For example, the discrete Fourier transformation is only faithful for Fourier transformation at certain frequencies, and it has a different degree of faithfulness for other frequencies. This is called the *unfaithful frequency response* [1] of the discrete Fourier transformation.

For each signal sequence $h[k], k = 0, 1 \dots N-1$, in which N is a positive integer, the discrete Fourier transformation $H[n], n = 0, 1 \dots N-1$ and its reverse transformation is given by Formula (9.1) and (9.2), respectively [11], in which k is the *index time* and n is the *index frequency* for the discrete Fourier transformation [18]

$$H[n] = \sum_{k=0}^{N-1} h[k]\ e^{i2\pi kn/N}; \tag{9.1}$$

$$h[k] = \frac{1}{N} \sum_{n=0}^{N-1} H[n]\ e^{-i2\pi nk/N}; \tag{9.2}$$

Formula (9.3) is the discrete forward transformation $H[n]$ of a pure sine signal $h[k] = \sin(2\pi kf/N)$ in which f is the signal frequency. The continuous forward transformation of the $h[k]$ is a delta function at $n = \pm f$ with phase $\pi/2$. $H[n]$ is delta-like function at $n = \pm f$ with phase $\pi/2$ only if f is an integer. In other cases, how much the result of discrete Fourier transformation deviates from continuous Fourier transformation depends on how much f deviates from an integer, e.g., when f is exactly between two integers, the phase of the transformation is that of cosine instead of sine according to Formula (9.3). Examples of unfaithful representations of fractional signal frequency by the discrete Fourier transformation are shown in Figure 40. The data for Figure 40 is generated using *SciPy*, which are very reproducible using any other math libraries, including *MathLab* and *Mathematica*. Figure 40 is a practical reproduction of a previous theoretical figure [1].

$$\begin{aligned} H[n] &= \sum_{k=0}^{N-1} \sin(2\pi nk/N)\ e^{i2\pi nk/N} = \frac{1}{2i} \left(\sum_{k=0}^{N-1} e^{i2\pi(n+f)\frac{k}{N}} - \sum_{k=0}^{N-1} e^{i2\pi(n-f)\frac{k}{N}} \right) \\ &= \begin{cases} iN/2, & f \text{ is integer} \\ N/\pi, & f \text{ is integer} + 1/2 \\ \frac{1}{2} \frac{\sin(2\pi f - 2\pi \frac{f}{N}) + \sin(2\pi \frac{f}{N}) - \sin(2\pi f) e^{-i2\pi \frac{n}{N}}}{\cos(2\pi \frac{n}{N}) - \cos(2\pi \frac{f}{N})} & \text{otherwise} \end{cases} \end{aligned} \tag{9.3}$$

Due to its width, a frequency component in an unfaithful transformation may interact with other frequency components of the Discrete Fourier spectrum, thus sabotaging the whole idea of using the Fourier Transformation to decompose a signal into independent frequency components. This means that the common method of

[18] The index frequency and index time are not necessarily related to time unit. The naming is just a convenient way to distinguish the two opposite domains in the Fourier transformation: the waveform domain vs the frequency domain.

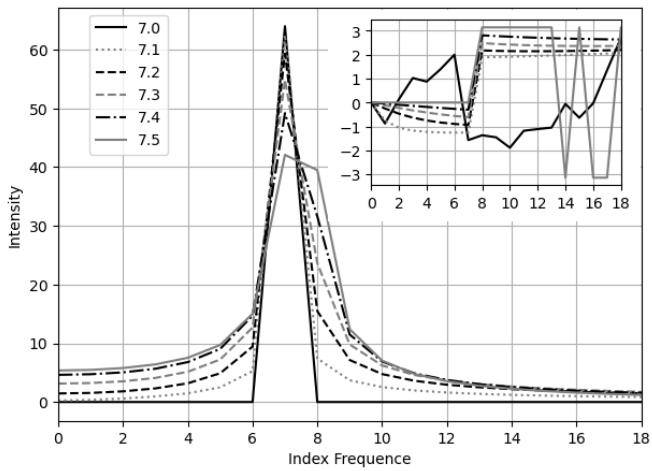

Figure 40: An unfaithful Discrete Fourier transformation is demonstrated by the spectra of a few sine signals having amplitude of 1 and slightly different frequencies as shown in legends. The x-axis is for index frequency. The y-axis is for intensity, while the y-axis of the embedded picture is for phase.

signal processing in the Fourier space [11][15][34] may generate artifacts due to its uniform treatment of faithful and unfaithful signal components, which probably coexist in reality. Unlike aliasing [11][15][37], unfaithful representation of the discrete Fourier transformation has an equal presence in the whole frequency range so that it cannot be avoided by sampling the original signal differently.

An unfaithful representation arises from the implied assumption of the discrete Fourier transformation. The continuous Fourier transformation has an infinitive signal range so that:

$$h(t) \Leftrightarrow H(s): \quad h(t-\tau) \Leftrightarrow H(s)e^{i2\pi s\tau}; \tag{9.4}$$

As an analog, the discrete Fourier transformation $G[n]$ of the signal $h[k], k = 1 \ldots N$ can be calculated mathematically from the discrete Fourier transformation $H[n]$ of $h[k], k = 0 \ldots N-1$:

$$G[n] = (H[n] + h[N] - h[0])e^{i2\pi n/N}; \tag{9.5}$$

Applying Formula (9.4) to Formula (9.5) results in Formula (9.6).

$$h[N] = h[0]; \tag{9.6}$$

Thus, the discrete Fourier transformation has an implied assumption that the signal $h[k]$ repeats itself periodically outside its region of $[0, N-1]$. For an unfaithful frequency, $h[N-1]$ and $h[N]$ are discontinuous in regard to signal periodicity, resulting in larger peak width, lower peak height, and wrong phase.

The unfaithfulness of the discrete Fourier transformation to the Fourier transformation is a very serious example of modeling errors, but this problem has not been addressed seriously enough previously. Because the discrete Fourier transformation has widest applications in science and engineering, this problem needs some serious attention.

9.2 FFT (Fast Fourier Transformation)

When $N = 2^L$, in which L is a positive integer, the generalized Danielson-Lanczos lemma [11] can be applied to the discrete Fourier transformation as FFT [11].

- For each output, each input is only used once, so there is no dependency problem when using Formula (2.5) and (2.9) as arithmetic operations.
- When L is large, the large amount of input and output data enables high quality statistical analysis.
- The amount of calculation is L, because for each output, increasing L by 1 results in one additional step of sum of multiplication.
- Each step in the forward transformation thus increases the variance by 2-fold, so that the result uncertainty mean increases with the FFT order L as $\sqrt{2}^L$. Because the reverse transformation divides the result by 2^L, the result uncertainty mean decreases with the FFT order L as $\sqrt{1/2}^L$. The result uncertainty means for the roundtrip transformations is thus: $\sqrt{2}^L \times \sqrt{1/2}^L = 1$.
- Forward and reverse transformations are identical except for a sign, so they are essentially the same algorithm, and their difference is purely due to input data.

Forward and reverse FFT transforms have data difference in normal usage:

- The forward transformation cancels real imprecise data of a Sin/Cos signal into a spectrum of mostly 0 values, so that both its value errors and its result uncertainties are expected to grow faster.
- The reverse transformation spread the spectrum of precise values of most 0 to data of a Sin/Cos signal, so that both its value errors and its result uncertainties are expected to grow slower.

9.3 Testing Signals

Only Formula (9.1) and (9.2) with integer n and k will be used.

The following signals are used for testing:

- *Sin*: $h[k] = \sin(2\pi kf/N), f = 1, 2, ...N/2 - 1$.
- *Cos*: $h[k] = \cos(2\pi kf/N), f = 1, 2, ...N/2 - 1$.
- *Linear*: $h[k] = k$, whose discrete Fourier transformation is Formula (9.7).

$$y \equiv i2\pi\frac{n}{N}: \quad G(y) = \sum_{k=0}^{N-1} e^{yk} = \frac{e^{Ny} - 1}{e^y - 1} = \begin{cases} y = 0: & N \\ y \neq 0: & 0 \end{cases} ;$$

$$H[n] = \frac{dG}{dy} = \begin{cases} n = 0: & \frac{N(N-1)}{2} \\ n \neq 0: & -\frac{N}{2}(1 + i/\tan(n\frac{\pi}{N})) \end{cases} ; \tag{9.7}$$

Empirically:

- The results from Sin and Cos signals are statistically indistinguishable from each other.
- The results from Sin signals at different frequencies are statistically indistinguishable from each other.

Thus, the results for Sin and Cos signals at all frequencies are pooled together for the statistical analysis, as the *Sin/Cos* signals.

9.4 Library Errors

Formula (9.1) and (9.2) limit the use of $\sin(x)$ and $\cos(x)$ to $x = 2\pi j/N$. To minimize the numerical errors of $\sin(x)$ and $\cos(x)$, *indexed sine functions* can be used instead of the library sine functions [19]

1. Instead of a floating-point value x for $\sin(x)$ and $\cos(x)$, the integer j is used to specify the input to $\sin(x)$ and $\cos(x)$, as $\sin(2\pi j/N)$ and $\cos(2\pi j/N)$, to avoid the floating-point rounding error of x.

2. The values of $\sin(2\pi j/N)$ and $\cos(2\pi j/N)$ are extended from $j = 0, 1, \ldots N/8$ to $j = 0, 1, \ldots N$ using the symmetry of $\sin(2\pi j/N)$ and $\cos(2\pi j/N)$.

3. The values of $\sin(2\pi j/N)$ and $\cos(2\pi j/N)$ are extended from $j = 0, 1, \ldots N$ to the whole integer region using the periodicity of $\sin(2\pi j/N)$ and $\cos(2\pi j/N)$.

Figure 41 shows that the value difference between the library $\sin(x)$ and the indexed $\sin(x)$ increases with increasing x. However, checked by $\sin(x)^2 + \cos(x)^2 - 1$, Figure 42 shows that the value errors are quite comparable between the indexed and library $\sin(x)$ and $\cos(x)$ functions. In Figure 42, the deviations of the value errors for the indexed sine function seem less stable statistically, because the deviations of the index sine function are calculated from much less samples: For the FFT order N, the deviations for the indexed sine function contain 2^{N-3} data covering $x \in [0, \pi/4]$, while the deviations for the library sine function contain 2^{2N-1} data covering $x \in [0, \pi 2^N]$. Figure 42 also shows that both $\sin(x)$ functions have proper coverage in variance arithmetic. Thus, the difference is just caused by the rounding of x in $\sin(x)$ and $\cos(x)$ library functions.

Figure 43 shows the value difference between the library $\cos(x)/\sin(x)$ and the indexed $\cos(x)/\sin(x)$ for $x \in (0, \pi)$. It covers all the integer inputs to the indexed cotan functions used for the Linear signal of FFT order 6 and 7. The value errors near $x = \pi$ increase with the FFT order. It is likely that the rounding error of x causes the numerical error when $\cos(x)/\sin(x)$ becomes very large near $x = \pi$. According to Figure 44, the library $\cos(x)/\sin(x)$ function does not have proper coverage in variance arithmetic.

Using indexed $\sin(x)$ as standard, Figure 44 compares the value errors of the library $\sin(x)$, $\cos(x)/\sin(x)$, and $1/tan(x)$ for increasing FFT orders. It shows that the numerical errors of all these library functions grow exponentially with increasing FFT order. $1/tan(x)$ has the largest errors, and it will not be used for FFT transformations. $\cos(x)/\sin(x)$ has larger errors than those of $\sin(x)$.

The difference between the indexed and library sine functions shows the inconsistency in the library sine functions, because the indexed sine functions is just the library sine functions in the range of $[0, \pi/4]$, and should be related to other ranges according to precise mathematical relations. It is difficult to study the numerical errors of library functions using conventional floating-point arithmetic which has no concept of imprecise values. Having identified the nature and the strength of the numerical errors of the library $\sin(x)$ and $\cos(x)/\sin(x)$, variance arithmetic can show their effects using FFT transformations.

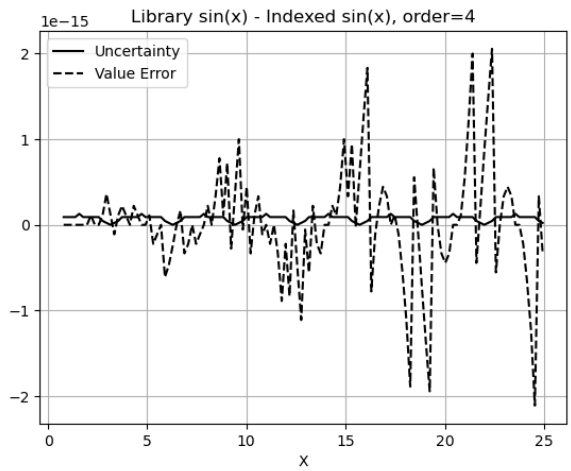

Figure 41: The difference between the library $\sin(x)$ and the indexed $\sin(x)$, for all integer input to the indexed sine functions used in the FFT transformations of FFT order 4. The uncertainties of the $\sin(x)$ values are also displayed, to mark the periodicity of π.

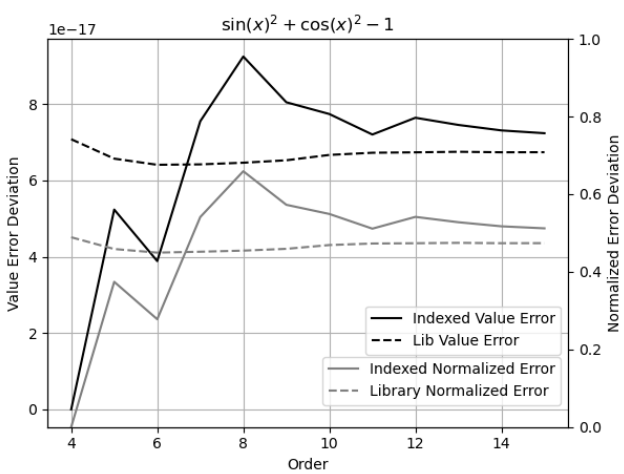

Figure 42: The value error deviation of $\sin(x)$ and $\cos(x)$ checked by $\sin(x)^2 + \cos(x)^2 - 1$ for different FFT order as shown in the x-axis, and for the indexed and library versions as shown in the legend. Also shown is the corresponding normalized error deviation for the indexed and library $\sin(x)$. The y-axis on the left is for value error deviation, while the y-axis on the right is for normalized error deviation.

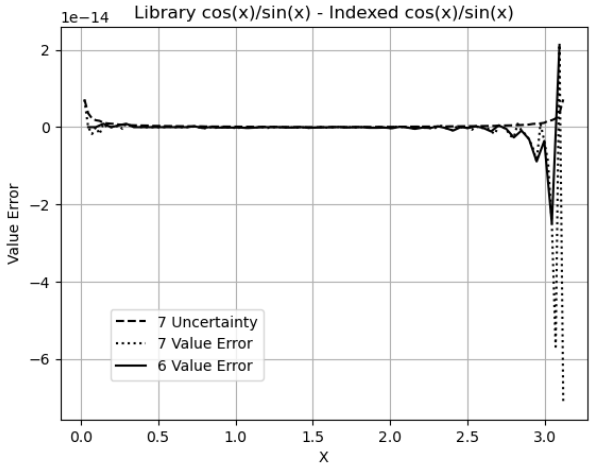

Figure 43: The difference between the library $\cos(x)/\sin(x)$ and the indexed $\cos(x)/\sin(x)$, for $x \in (0, \pi)$, for different FFT order as shown in the legend. In the legend, *Uncertainty* is the calculated uncertainty assuming that both $\cos(x)$ and $\sin(x)$ are imprecise in their least significant values, and *Value Error* is the difference between the library $\cos(x)/\sin(x)$ and the indexed $\cos(x)/\sin(x)$.

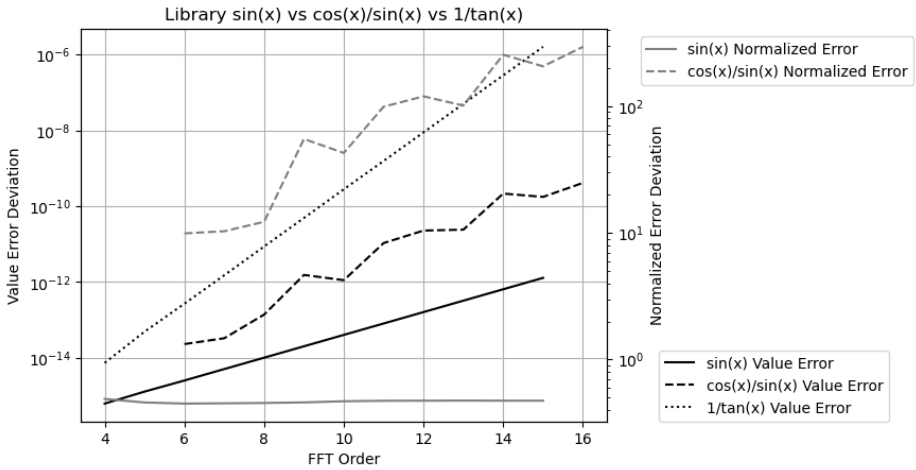

Figure 44: Comparing library $\sin(x)$ and $\cos(x)/\sin(x)$ for different FFT orders as shown by x-axis, and for either value error deviations or normalized error deviations, as shown in the legend. The y-axis on the left is for value error deviations, while the y-axis on the right is for normalized error deviations.

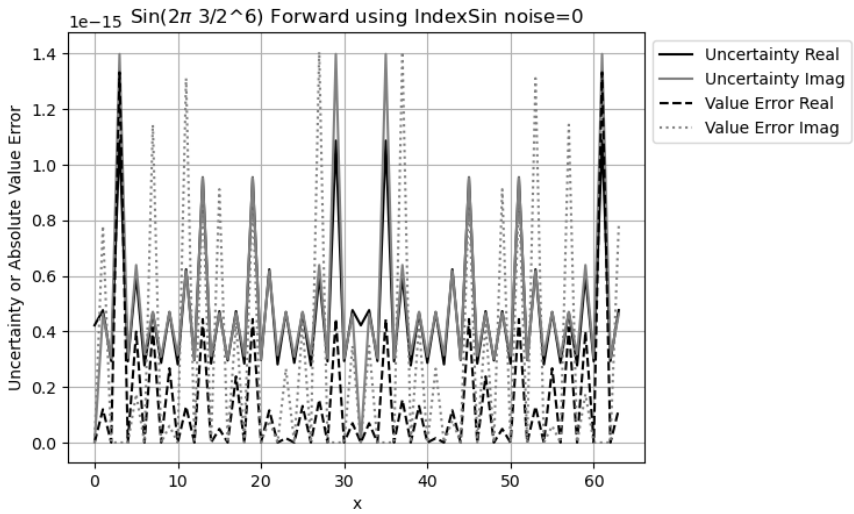

Figure 45: The FFT spectrum of $\sin(j3/2^6\pi)$ using the indexed sine functions after the forward transformation calculated by variance arithmetic, with the uncertainty and the value errors shown in the legend. The x-axis is for index frequency. The y-axis is for uncertainty and absolute value error.

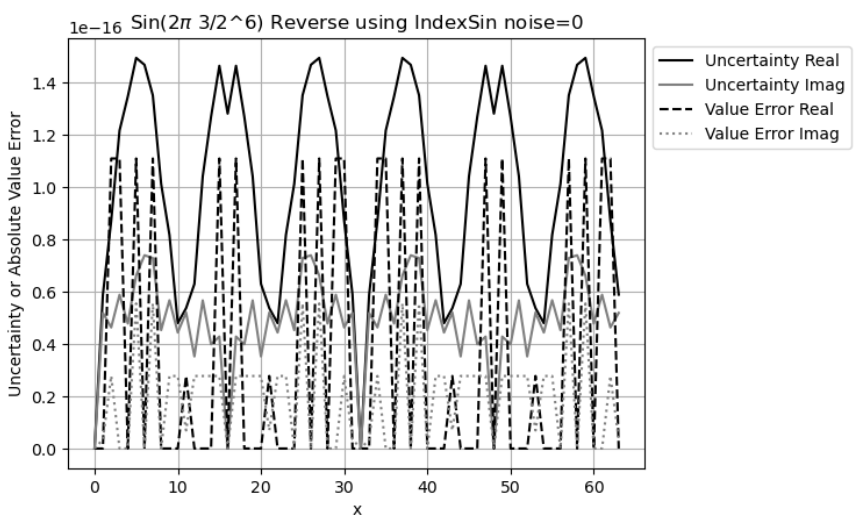

Figure 46: The FFT waveform of $\sin(j3/2^6\pi)$ using the indexed sine functions after the reverse transformation calculated by variance arithmetic, with the uncertainty and the value errors shown in the legend. The x-axis is for index time. The y-axis is for uncertainty and absolute value error.

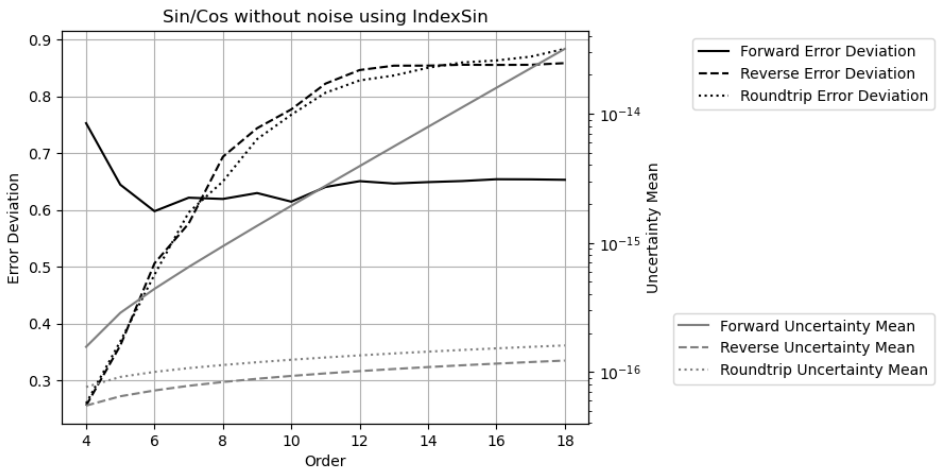

Figure 47: The result error deviation and uncertainty mean of Sin/Cos signals vs. FFT order using the indexed sine functions for forward, reverse and roundtrip FFT transformations, as shown in the legend. The y-axis on the left is for error deviation, while the y-axis on the right is for uncertainty mean.

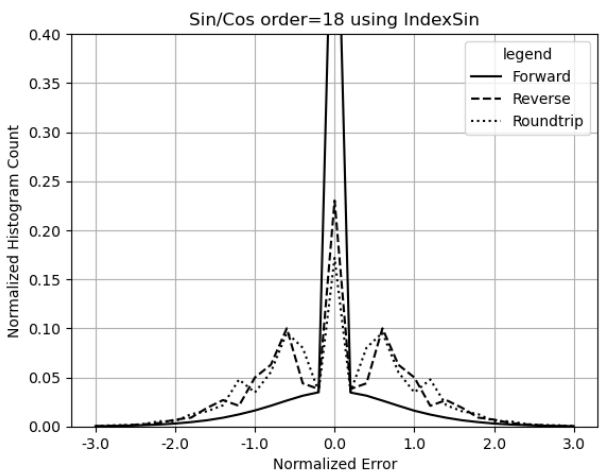

Figure 48: The histograms of the normalized errors of Sin/Cos signals using the indexed sine functions for forward, reverse and roundtrip FFT transformations, as shown in the legend. The FFT order is 18.

9.5 Using the Indexed Sine Functions for Sin/Cos Signals

Using the indexed sine functions, for the waveform of $\sin(2\pi k3/2^6)$ in which k is the index time:

- Figure 45 shows that for the forward transformation, the result value errors are comparable to the result uncertainties, with an error deviation of 0.37 for the real part, and 0.93 for the imaginary part. Both real and imaginary uncertainties peak at the expected index of ± 3, but only real value error peak at the frequency. Both real and imaginary uncertainties peak at the unexpected index of ± 29, ± 13 and ± 18, as well as the value errors. The reason for the unexpected peaks is not clear. Overall, the uncertainty seems a smoothed version of the value error, and they are comparable in strength.
- Figure 46 shows that for the reverse transformation, the result value errors are comparable to the result uncertainties, with an error deviation of 0.54 for the real part, and 0.52 for the imaginary part. The uncertainty captures the periodicity of the value error, as well as bigger strength of the real value error than that of the imaginary value error.

Figure 47 shows both the result uncertainty means and the error deviations vs. FFT order of FFT transformations of Sin/Cos signals using the indexed sine functions. As expected, the uncertainties grow much faster with increasing FFT order for the forward transformation. The error deviations are a constant about 0.65 for the forward transformation, while they increase with increasing FFT order from 0.25 until reach a constant about 0.85 when the FFT order is 12 or more for the reverse and roundtrip transformation.

With increasing FFT order, all histograms become more Gaussian like, and the error deviations for the real and the imaginary parts become more equal in value. Figure 48 shows the corresponding histograms of the normalized errors at FFT order 18. The distribution of the forward transformation is a larger Delta distribution on top of a Gaussian distribution, while it is a structured distribution on top of a Gaussian distribution for the reverse transformation. The distribution of the roundtrip transformation is almost identical to that of the reverse transformation, suggesting that the presence of input uncertainty for the reverse transformation is not a major factor.

Proper coverage is achieved using the indexed sine functions for Sin/Cos signals.

9.6 Using the Library Sine Functions for Sin/Cos Signals

Because the least significant values are the only source of input uncertainties for variance arithmetic, the result uncertainties using either sine functions are almost identical. The library sine functions have numerical calculation errors which is not specified by the input uncertainties of variance arithmetic. The question is whether variance arithmetic can track these numerical errors effectively or not.

Using the library sine functions, for the waveform of $\sin(2\pi k3/2^6)$ in which k is the index time:

- Figure 49 shows that for the forward transformation, the result value errors are noticeably larger than the result uncertainties, with an error deviation of 7.0 for

[19] The sin and cos library functions in Python, C++, Java all behave in the same way. The indexed sine functions are similar to *sinpi* and *cospi* functions proposed in C++23 standard but not readily available excepted in few selected platforms such as for Apple.

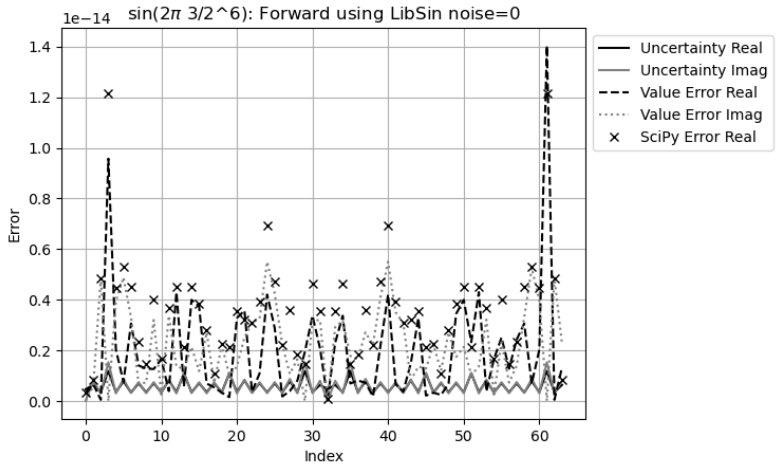

Figure 49: The FFT spectrum of $\sin(j3/2^6\pi)$ using the library sine functions after the forward transformation calculated by variance arithmetic, with the uncertainties and the value errors shown in the legend. Also included are the corresponding result value errors using *SciPy*. The x-axis is for index frequency. The y-axis is for uncertainties or absolute value errors.

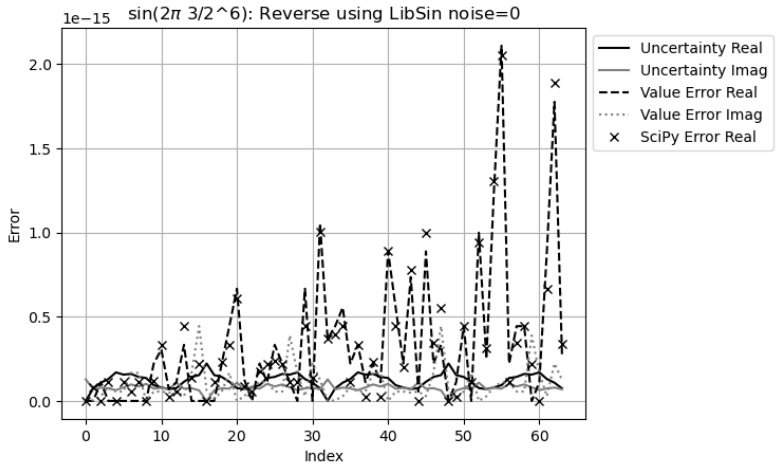

Figure 50: The FFT waveform of $\sin(j3/2^6\pi)$ using the indexed sine functions after the reverse transformation calculated by variance arithmetic, with the uncertainties and the value errors shown in the legend. Also included are the corresponding result value errors using *SciPy*. The x-axis is for index time. The y-axis is for uncertainties or absolute value errors.

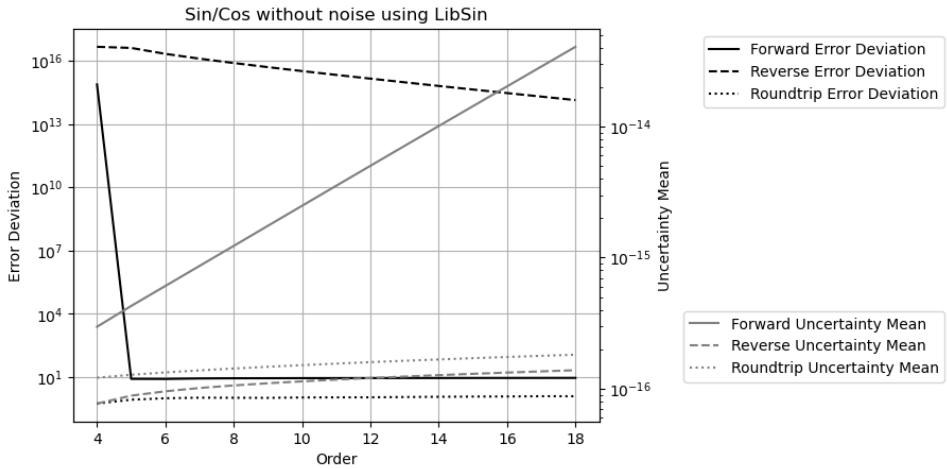

Figure 51: The result error deviation and uncertainty mean of Sin/Cos signals vs. FFT order using the library sine functions for forward, reverse and roundtrip FFT transformations, as shown in the legend. The y-axis on the left is for error deviation, while the y-axis on the right is for uncertainty mean.

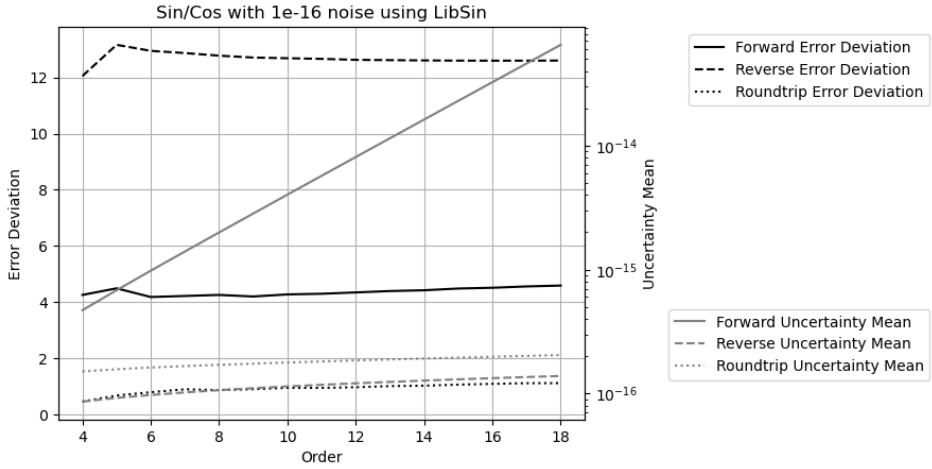

Figure 52: The result error deviation and uncertainty mean of Sin/Cos signals vs. FFT order using the library sine functions for forward, reverse and roundtrip FFT transformations, as shown in the legend. The y-axis on the left is for error deviation, while the y-axis on the right is for uncertainty mean. 10^{-16} input noises are added to the Sin/Cos signals.

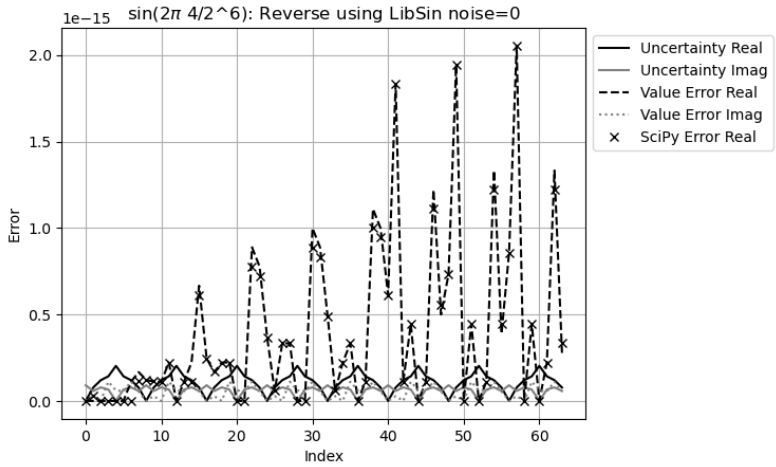

Figure 53: The waveform of $\sin(j4/2^6\pi)$ using the indexed sine functions after the reverse transformation calculated by variance arithmetic, with the uncertainties and the value errors shown in the legend. Also included are the corresponding result value errors using *SciPy*. The x-axis is for index time. The y-axis is for uncertainties or absolute value errors.

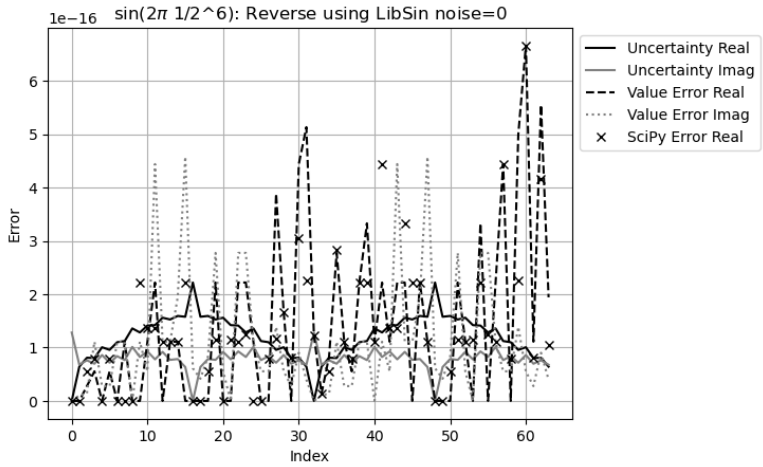

Figure 54: The waveform of $\sin(j1/2^6\pi)$ using the indexed sine functions after the reverse transformation calculated by variance arithmetic, with the uncertainties and the value errors shown in the legend. Also included are the corresponding result value errors using *SciPy*. The x-axis is for index time. The y-axis is for uncertainties or absolute value errors.

the real part, and 5.6 for the imaginary part. As expected, the error deviations are noticeably larger than their counterparts in Figure 45.

- Figure 50 shows that for the reverse transformation, the result value errors are noticeably larger than the result uncertainties, with an error deviation of 5.7 for the real part, and 0.83 10^{15} for the imaginary part. The surprisingly large imaginary error deviation is caused by the small uncertainty at the index time of 8 where the value error is at a local maximum.
- Because of the result of the huge error deviation, Figure 51 shows that the result of the reverse transformation no longer has proper coverage for all FFT orders. When a small noise with deviation of 10^{-16} is added to the input, the result uncertainty of the reverse transformation at the index time of 8 is no longer near 0, so that the result error deviations achieve proper coverage again, as show in Figure 52.
- Figure 47 and Figure 51 confirm that the result uncertainties are identical to all FFT transformations, respectively.

Variance arithmetic reveals the reason why to add small noises to the numerically generated input data, which is already a common practice [11]. It further reveals that the amount of noise needed is to achieve proper coverage.

In Figure 50, the value errors tend to increase with the index time, which is due to the periodic increase of the numerical errors of $\sin(x)$ with x as shown in Figure 41. When the signal frequency increases, the increase of the value errors with the index frequency in the reverse transformation becomes stronger, as shown in Figure 50, 53, and 54. In fact, Figure 53 suggests that the periodic numerical errors in Figure 41 resonate with the periodicity of the input *Sin* signal. In contrast, Figure 46 shows no indication for such increase. It shows that the library numerical errors may have surprisingly large effect on the numerical results.

To validate the FFT implementation in variance arithmetic, the results are compared with the corresponding calculations using the python numerical library *SciPy* in Figure 49 and 50. The results are quite comparable, except that in *SciPy*, the data waveform is always real rather than complex. In Figure 49, the *SciPy* results have been made identical for the frequency indexes f and $-f$, and the matching to the corresponding results of variance arithmetic is moderate. In Figure 50, the *SciPy* results match the corresponding results of variance arithmetic quite well. It shows that the effects of library numerical errors revealed by variance arithmetic exist in real applications.

9.7 Linear Signal

As shown in Figure 44, linear signals may introduce more numerical errors to the result through library $\cos(x)/sin(x)$. The question is whether variance arithmetic can track these additional numerical errors or not.

Using the indexed sine functions:

- Figure 55 shows that the result uncertainty for each FFT transformation has much larger increase with the increasing FFT order than its counter part in Figure 47. The much faster uncertainty increase of the forward transformation is attributed to the non-zero input uncertainties. Figure 55 shows that proper coverage can be achieved for all FFT transformations for all FFT orders.

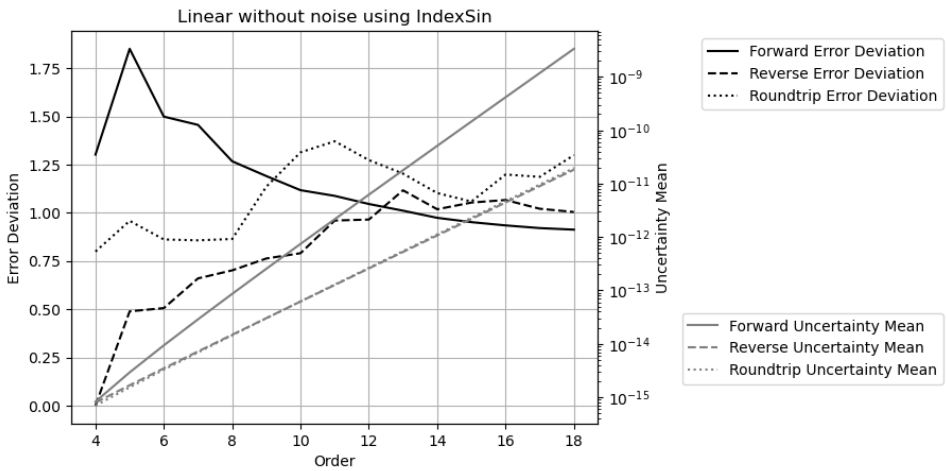

Figure 55: The result error deviation and uncertainty mean of Linear signal vs. FFT order using the indexed sine functions for forward, reverse and roundtrip FFT transformations, as shown in the legend. The y-axis on the left is for error deviation, while the y-axis on the right is for uncertainty mean.

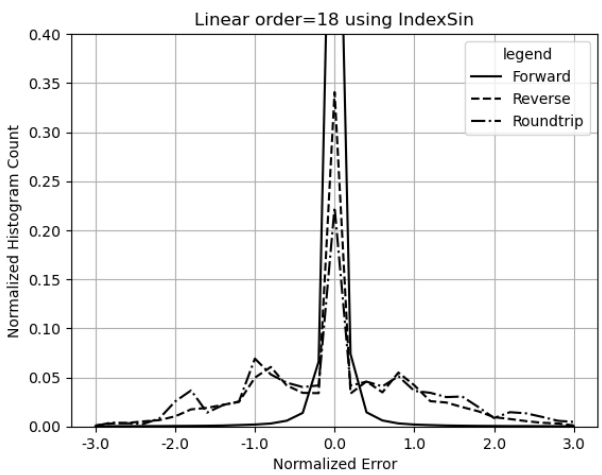

Figure 56: The histograms of the normalized errors of Linear signal using the indexed sine functions for forward, reverse and roundtrip FFT transformations, as shown in the legend.

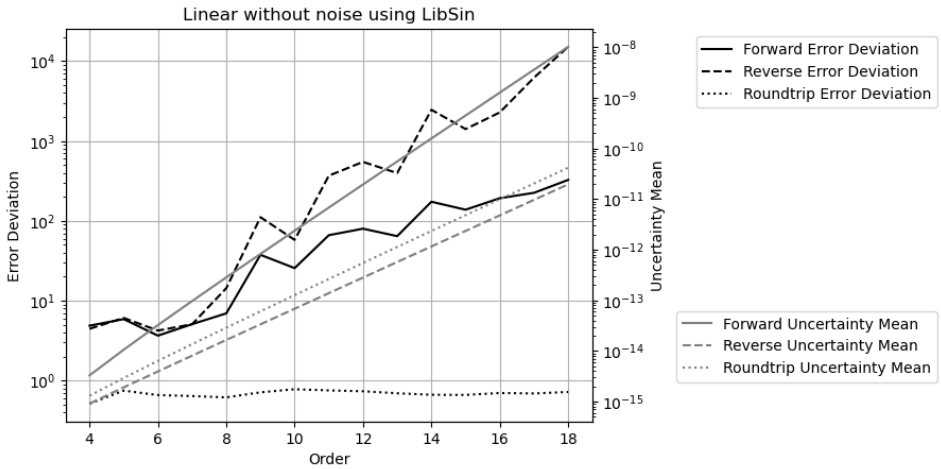

Figure 57: The result error deviation and uncertainty mean of Linear signal vs. FFT order using the library sine functions for forward, reverse and roundtrip FFT transformations, as shown in the legend. The y-axis on the left is for error deviation, while the y-axis on the right is for uncertainty mean.

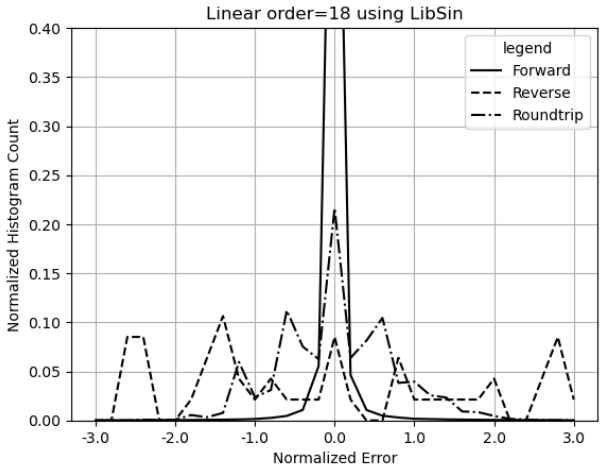

Figure 58: The histograms of the normalized errors of Linear signal using the library sine functions for forward, reverse and roundtrip FFT transformations, as shown in the legend.

- Figure 56 shows that for each transformation, the histogram of the Linear signal is somewhat similar to the counterpart of the Sin/Cos signals in Figure 48, except that the histograms for linear signal have larger Gaussian background.

Using the library sine functions:

- Figure 57 shows that proper coverage cannot be achieved for all FFT transformations for all FFT orders, because the value errors outpace the uncertainties with increasing FFT order, to result in worse error deviations for increasing FFT order.
- Figure 58 shows that the reverse histogram contains additional peaks to its counterpart in Figure 56. The additional peaks extend quite beyond the range of $[-3, +3]$, revealing the reason why variance arithmetic can not cover these value errors. Such large numerical errors are expected from Figure 44.

The result difference using variance arithmetic for library Sin/Cos signal vs library Linear signal suggests that variance arithmetic can break down when the input contains too much unspecified errors. The only solution seems to create a new sin library using variance arithmetic so that all numerical calculation errors are accounted for.

9.8 Ideal Coverage

Adding enough noise to the input can overpower unspecified input errors, to achieve ideal coverage, such as 10^{-3} input noise for Linear signal using the library sine functions.

- Figure 59 shows the corresponding histogram. As expected, the normalized errors for the forward and the reverse transformations are Normal distributed, even when the input noise is not Gaussian. The normalized errors for the roundtrip transformations are Delta distributed at 0, meaning the input uncertainties are perfectly recovered.
- Figure 60 show the corresponding error deviations and uncertainty means:
  - As expected, the result uncertainty means for the forward transformations increase with the FFT order L as $\sqrt{2}^L$.
  - As expected, the result uncertainty means for the reverse transformations decrease with the FFT order L as $\sqrt{1/2}^L$.
  - As expected, the result uncertainty means for the roundtrip transformations always equal the corresponding input uncertainties of 10^{-3}.
  - As expected, the result error deviations for the forward and reverse transformations are constant 1, while the result error deviations for the roundtrip transformation approaches 0 exponentially with increasing FFT order.

Also, the result uncertainty means for both the forward and the reverse transformations are linear to the input uncertainties, respectively, which is expected because FFT transformations are linear.

The range of ideal coverage depends on how well the input uncertainty specifies the input noise. For Linear signals using library sine functions, Figure 61 and 62 show the error deviation vs. the added noise vs. FFT order for the forward and reverse transformations, respectively. The ideal coverage is shown as the regions where the error deviations are 1. In other regions, proper coverage is not achievable. Because

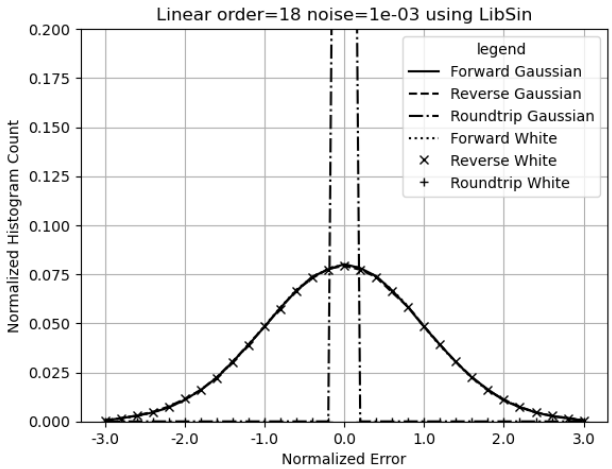

Figure 59: The histograms of the normalized errors of Linear signal with 10^{-3} input noise using the library sine functions for forward, reverse and roundtrip FFT transformations, as shown in the legend.

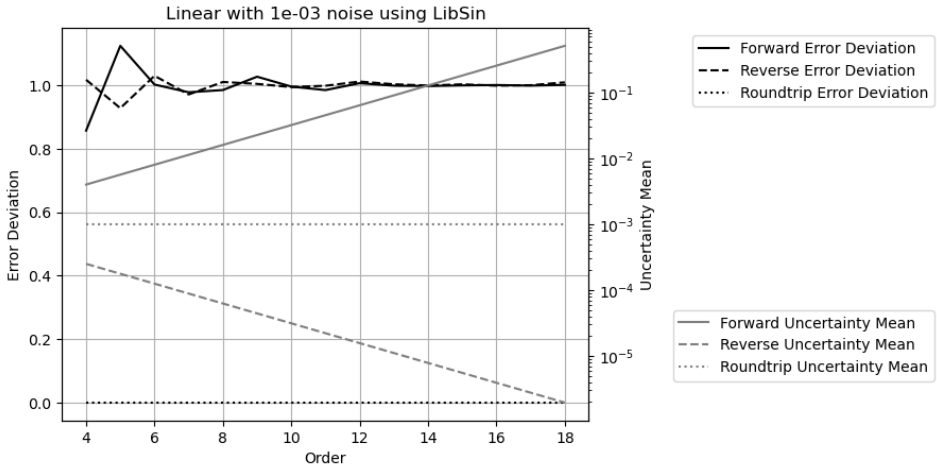

Figure 60: The result error deviation and uncertainty mean of Linear signal with 10^{-3} input noise using the library sine functions vs. FFT order for forward, reverse and roundtrip FFT transformations. The y-axis on the left is for error deviation, while the y-axis on the right is for uncertainty mean.

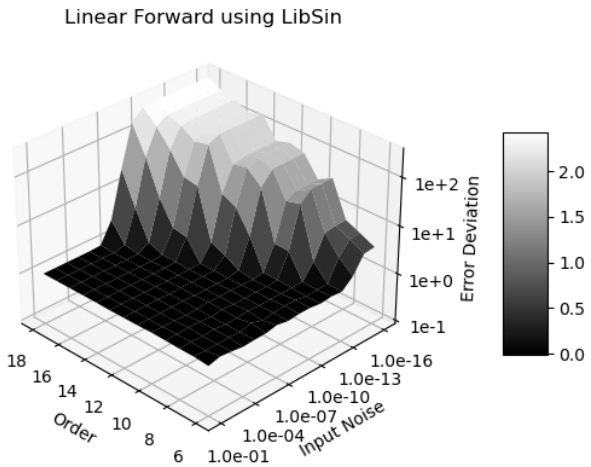

Figure 61: The result error deviations for Linear signals using library sine functions vs. input uncertainties and FFT orders for the forward transformations. The input uncertainties run from 10^{-16} to 10^{-1}, while the FFT Order runs from 6 to 18.

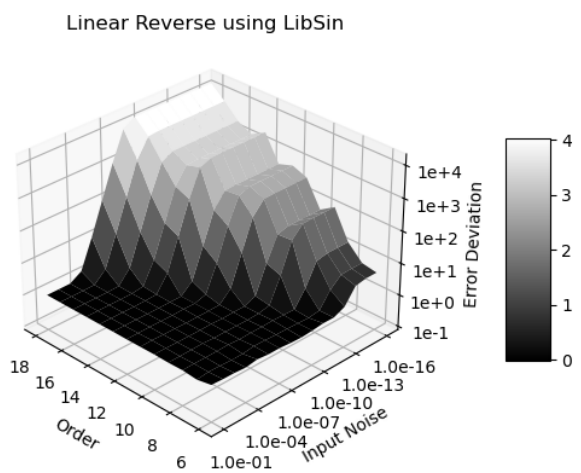

Figure 62: The result error deviations for Linear signals using library sine functions vs. input uncertainties and FFT orders for the reverse transformations. The input uncertainties run from 10^{-16} to 10^{-1}, while the FFT Order runs from 6 to 18.

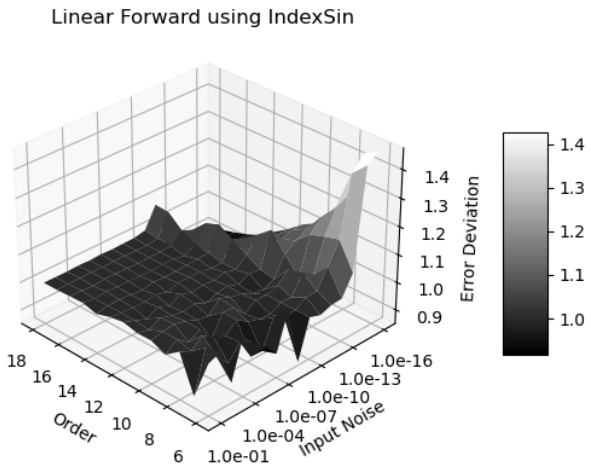

Figure 63: The result error deviations for Linear signals using indexed sine functions vs. input uncertainties and FFT orders for the forward transformations. The input uncertainties run from 10^{-16} to 10^{-1}, while the FFT Order runs from 6 to 18.

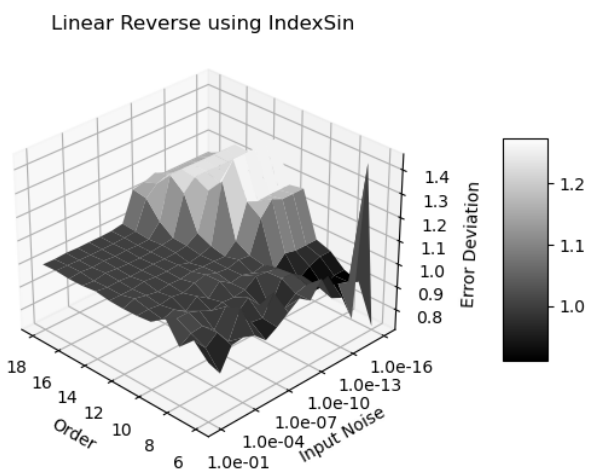

Figure 64: The result error deviations for Linear signals using indexed sine functions vs. input uncertainties and FFT orders for the reverse transformations. The input uncertainties run from 10^{-16} to 10^{-1}, while the FFT Order runs from 6 to 18.

the uncertainties grow slower in the reverse transformation than in the forward transformation, the reverse transformations have smaller ideal coverage region than that of the forward transformation. Because numerical errors increase with the amount of calculation, the input noise range for ideal coverage reduces with increasing FFT order. It is possible that ideal coverage is not achievable at all, e.g., visually, when the FFT order is larger than 25 for the reverse transformation. As one of the most robust numerical algorithms that is very insensitive to input errors, FFT can breaks down due to the numerical errors in the library sine functions, and such deterioration of the calculation result is not easily detectable using the conventional floating-point calculation.

In contrast, for Linear signals using indexed sine functions, as shown by Figure 63 and 64 for the forward and reverse transformations, respectively, the ideal region is much larger, and proper coverage is achieved in other regions. Forward FFT transformation still shows larger region of ideal coverage than that of Reverse FFT transformation.

For Sin/Cos using either indexed sine functions or library sine functions, the ideal region is achieved when the added noise is large enough to cover the effect of rounding errors, almost independent of FFT orders. In both cases, forward transformation needs 10^{-15} added noise, while reverse transformation needs 10^{-12} added noise. The difference is that using indexed sine functions, in the proper coverage region, error deviations deviate from 1 only slightly.

9.9 Summary

Compared to its counterpart continuous Fourier Transformation (FT), the discrete Fourier transformation (DFT) has large modeling error due to its implied assumption. This modeling error has not been addressed seriously.

The library sine functions using conventional floating-point arithmetic have been shown to contain numerical errors as large as equivalently 10^{-3} of input precision for FFT transformations. The library $\sin(x)$ errors increase periodically with x, causing noticeable increase of the result errors with increasing index frequency in the reverse transformation, even in resonant fashion. The dependency of the result numerical errors on the amount of calculation and input data means that a small-scale test cannot properly qualify the result of a large-scale calculation. The effect of numerical errors inside math library has not been addressed seriously.

Variance arithmetic should be used, whose values largely reproduce the corresponding results using conventional floating-point arithmetic, whose uncertainties trace all input errors, and whose result error deviations qualify the calculation quality as either ideal, or proper, or suspicious. If the library functions also have proper coverage, the calculation result will probably have proper coverage as well.

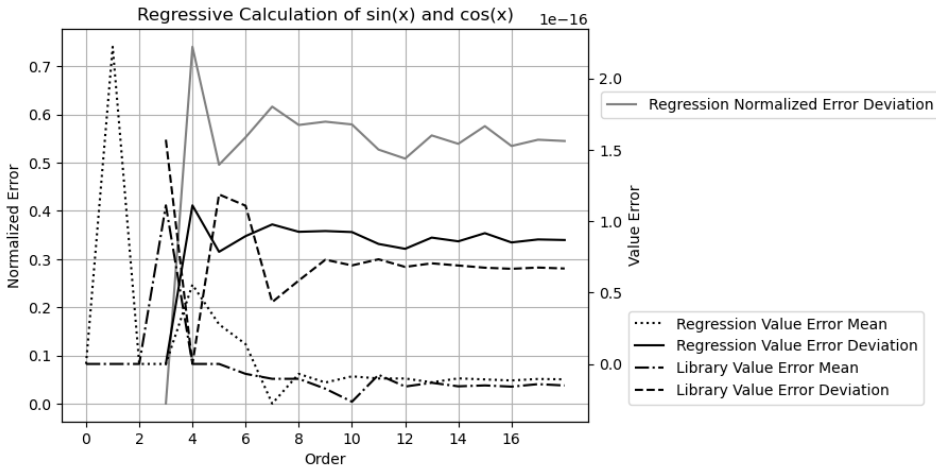

Figure 65: The error deviation and mean for $\sin(x)^2 + \cos(x)^2 - 1, x \in [0, \pi/4]$, for different regression order as shown by the x-axis. In the legend, *Value Error* means the values of $\sin(x)^2 + \cos(x)^2 - 1$, *Normalized Error* means the value errors normalized by the calculated uncertainty, *Regression* means the $\sin(x)$ and $\cos(x)$ generated by regression, and *Library* means library $\sin(x)$ and $\cos(x)$. The y-axis on the left is for normalized error, while the y-axis on the right is for value error.

10 Regressive Generation of Sin and Cos

Starting from Formula (10.1), Formula (10.2) and Formula (10.3) can be used recursively to calculate the sin library function.

$$\sin(0) = \cos(\frac{\pi}{2}) = 0; \qquad \sin(\frac{\pi}{2}) = \cos(0) = 1; \quad (10.1)$$

$$\sin\left(\frac{\alpha+\beta}{2}\right) = \sqrt{\frac{1-\cos(\alpha+\beta)}{2}} = \sqrt{\frac{1-\cos(\alpha)\cos(\beta)+\sin(\alpha)\sin(\beta)}{2}}; \quad (10.2)$$

$$\cos\left(\frac{\alpha+\beta}{2}\right) = \sqrt{\frac{1+\cos(\alpha+\beta)}{2}} = \sqrt{\frac{1+\cos(\alpha)\cos(\beta)-\sin(\alpha)\sin(\beta)}{2}}; \quad (10.3)$$

When $\alpha + \beta \to 0$, the nominator in Formula (10.2) is the difference of two values both are very close to 1, to result in very coarse precision. Also, both value errors and uncertainties are accumulated in the regression. This regression is not suitable for calculating library sin and cos functions. It can be used to check the trigonometric relation for library sin and cos functions.

The number of regressions is defined as the order. For each order n, 2^n $\sin(\frac{\pi}{2}\frac{i}{2^n})$ and $\cos(\frac{\pi}{2}\frac{i}{2^n})$ values are obtained, so that the result is more statistically stable with increasing n.

The value errors of $\sin(x)$ and $\cos(x)$ are checked by $\sin(x)^2+\cos(x)^2-1$. Figure 65 shows that the value errors of the generated $\sin(x)$ and $\cos(x)$ is comparable to those of library $\sin(x)$ and $\cos(x)$. It shows that the result error deviations are close to 0.5.

Because floating-point rounding errors are the only source of errors in the regression, as expected, variance arithmetic provides proper coverage for this regression.

11 Conclusion and Discussion

11.1 Summary of Variance Arithmetic

The starting point of variance arithmetic is the uncorrelated uncertainty assumption of all input variables. It is a very reasonable statistical requirement on the input data [1].

Once the uncorrelated uncertainty assumption is satisfied, variance arithmetic quantifies uncertainty as the deviation of the value errors. It can trace the variable dependency in the intermediate steps using standard statistics. The statistical nature of variance arithmetic is reflected by the bounding leakage $\epsilon = 5.73\ 10^{-7}$ and the required value precision $\tau = 7.18\ 10^{-7}$, which determine the convergence and truncation when Taylor expansion is used to obtain the result mean and uncertainty of an analytic function. In variance arithmetic, ill-formed problems can be invalidated as result divergence, unreliable result variance, or being practically unstable.

The presence of ideal coverage is the necessary condition for a numerical algorithm in variance arithmetic to be correct. The ideal coverage also defines the ideal applicable range for the algorithm. In ideal coverage, the calculated uncertainty equals the error deviation, and the result normalized errors is either Normal distributed or Delta distributed depending on the context. In variance arithmetic, the best-possible precision to achieve ideal coverage is 10^{-15}, which is good enough for most applications.

Variance arithmetic can only provide proper coverage for floating-point rounding errors, with the empirical error deviations in the approximate range of $[1/3, 3]$. Table 3 shows the measured error deviations in different contexts.

Variance arithmetic has been showcased to be widely applicable, so as for analytic calculations, progressive calculation, regressive generalization, polynomial expansion, statistical sampling, and transformations.

The code and analysis for variance arithmetic are published as an open source project at `https://github.com/Chengpu0707/VarianceArithmetic`.

Context	Error Deviation
Polynomial Taylor Expansion	2
Adjugate Matrix	4
$\sqrt[p]{x^p} - x$	0.55
Forward FFT of Sin/Cos signal with indexed sine	0.65
Reverse FFT of Sin/Cos signal with indexed sine	0.85
Roundtrip FFT of Sin/Cos signal with indexed sine	0.87
Forward FFT of Linear signal with indexed sine	0.8
Reverse FFT of Linear signal with indexed sine	1.0
Roundtrip FFT of Linear signal with indexed sine	1.25
Regressive Calculation of sin and cos	0.55

Table 3: The error deviations for floating-point rounding errors in different contexts.

11.2 Improvement Needed

This paper just showcases variance arithmetic which is still in its infancy. Theoretically, some important questions remain:

- Is there a way for variance arithmetic to provide ideal coverage to floating-point rounding errors? Many theoretical calculations have no input uncertainties, so that when the generation mechanism for rounding error is thoroughly understood, a special version of variance arithmetic for ideal coverage to floating-point rounding errors may be desirable.
- What is the uncertainty upper bounds for each problem? Specifically, how to explain quantitatively the measured upper bounds in Figure 25 and 31.
- How to analytically quantify the reliability requirements for variance arithmetic?
- How to apply variance arithmetic when analytic solution is not available such as solving differential equation?

11.3 New Numerical Approaches

Traditional numerical approaches focus primarily on the result values, and they need to be reexamined or even reinvented in variance arithmetic. Most traditional numerical algorithms try to choose one of the best paths, while variance arithmetic rejects conceptually all path-dependent calculations.

Variance arithmetic generally reveals worse results than those of conventional floating-point arithmetic, such as very quick loss of precision in matrix inversion, especially for matrix of large size. How to reconcile variance arithmetic and traditional numerical approaches could be a big challenge.

Variance arithmetic no longer has dependency problem, and it can reject ill-formed problems. By using stability truncation, variance arithmetic no longer needs theoretical reminder estimator so that it is simpler for truncation problems. Variance arithmetic seems conceptually easier than traditional numerical approaches.

The calculation of the result uncertainty bias and uncertainty contains a lot of sums of constant terms such as Formula (2.18) and (2.50), so that it is a excellent candidate for parallel processing. Because variance arithmetic has neither execution freedoms nor the associated dependency problems, it can be implemented either in software using GPU or directly in hardware.

In variance arithmetic, it is generally an order-of-magnitude more complicated to calculate the result uncertainties than the result values, such as Formula (5.19) for Taylor expansion of matrix inversion. However, modern software programs for analytic calculation such as *SymPy* can be a great help, whose usage is demonstrated in the corresponding open source project. Perhaps it is time to move from numerical programming to analytic programming.

11.4 Recalculating Library Math Functions

The library math functions need to be calculated using variance arithmetic, so that each output value has its corresponding uncertainty. Otherwise, the effect of the existing value errors in the library function can have unpredictable and large effects. For example, this paper shows that the periodic numerical errors in the sine library functions cause resonant-like large result value errors in a FFT reverse transformation.

11.5 Different Floating-Point Representation for Variance

In variance representation $x \pm \delta x$, δx is comparable in value with x, but $(\delta x)^2$ is calculated and stored. This limits the effective range for $x \pm \delta x$ to be much smaller

than the full range of the standard 64-bit floating-point representation. Ideally, $(\delta x)^2$ should be calculated and stored in an unsigned 64-bit floating-point representation in which the sign bit reused for the exponent, so that δx has exactly the same range as the standard 64-bit floating-point representation.

11.6 Uncertainty of Variance

In Formula (2.18) and (2.50), the result variance is calculated using variance arithmetic:

- If the uncertainty of the variance is more than $1/\sigma$-fold of the value of the variance, the result is deemed to be unreliable;
- Otherwise, the uncertainty of the variance is ignored.

However, the above approach is invalid if the accumulation of the uncertainty of variance cannot be ignored.

On the other hand, if the uncertainty of variance is traced, how about the uncertainty of the uncertainty of the variance, etc.? Thus, the theoretical foundation of variance arithmetic needs to be improved.

11.7 Lower Order Approximation

Most practical calculations are neither pure analytic nor pure numerical, in which the analytic knowledge guides the numerical approaches, such as solving differential equations. In these cases, when input precision is fine enough, lower-order approximation of variance arithmetic may be a viable solution because these calculations may only have low-order derivatives for the result. This paper provides an example of first order approximation of variance arithmetic, as Formula (5.21) for $\delta^2|\mathbf{M}|$. According to Section 2, the convergence and stability of the result has to be assumed for lower-order approximations. How to effectively apply variance arithmetic to practical calculations remains an open question.

11.8 Conceptual Extension to Correlation

Although the first-order approximations of Formula (2.17) and has been published relatively recently [38], it is safe to say that the effect of input uncertainty to the result value has not been taken seriously.

The first-order approximations of Formula (2.50) has been explained in the traditional variance-covariance framework [38], which only considered linear interaction between dependence variables through an analytic mathematical function, while Formula (2.50) is a more general quantification of interaction between dependence variables through the analytic mathematical function.

11.9 Source Tracing

As an enhancement for dependency tracing, source tracing points out the contribution to the result uncertainty bias and uncertainty from each input. This gives clues to engineers on how to improve a measurement effectively.

11.10 Variable σ Variance Arithmetic

Variance arithmetic rejects a calculation if its distributional pole or distributional zero is within the input ranges. But there are many use cases in which such intrusion cannot be avoided, so variance arithmetic of smaller σ may be needed. It is even possible in another implementation of variance arithmetic, σ is part of a variance representation, so that a calculation closer to a distributional zero or a distributional pole can proceed but at the expense of reducing σ. The result σ for a calculation of imprecise inputs each with different σ is found statistically through the corresponding ϵ. Such approaches need theoretical foundation in both mathematics and statistics.

11.11 Acknowledgments

As an independent researcher without any academic association, the author of this paper feels indebted to encouragements and valuable discussions with Dr. Zhong Zhong from Brookhaven National Laboratory, Prof Weigang Qiu from Hunter College, the organizers of *AMCS 2005*, with Prof. Hamid R. Arabnia from University of Georgia in particular, and the organizers of *NKS Mathematica Forum 2007*, with Dr. Stephen Wolfram in particular. Prof Dongfeng Wu from Louisville University provides valuable guidance and discussions on statistical related topics. The author of this paper is very grateful for the editors and reviewers of *Reliable Computing* for their tremendous help in shaping and accepting the previous version of this paper from unusual source, with managing editor, Prof. Rolph Baker Kearfott in particular.

References

[1] C. P. Wang. A new uncertainty-bearing floating-point arithmetic. *Reliable Computing*, 16:308–361, 2012.

[2] Sylvain Ehrenfeld and Sebastian B. Littauer. *Introduction to Statistical Methods*. McGraw-Hill, 1965.

[3] John R. Taylor. *Introduction to Error Analysis: The Study of Output Precisions in Physical Measurements*. University Science Books, 1997.

[4] Jurgen Bortfeldt, editor. *Fundamental Constants in Physics and Chemistry*. Springer, 1992.

[5] John P Hayes. *Computer Architecture*. McGraw-Hill, 1988.

[6] David Goldberg. What every computer scientist should know about floating-point arithmetic. *ACM Computing Surveys*, March 1991.

[7] Institute of Electrical and Electronics Engineers. *ANSI/IEEE 754-2008 Standard for Binary Floating-Point Arithmetic*, 2008.

[8] U. Kulish and W.M. Miranker. The arithmetic of digital computers: A new approach. *SIAM Rev.*, 28(1), 1986.

[9] J. H. Wilkinson. *Rounding Errors in Algebraic Processes*. SIAM, 1961.

[10] Nicholas J. Higham. *Accuracy and Stability of Numerical Algorithms*. SIAM, 2002.

[11] William H. Press, Saul A Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C*. Cambridge University Press, 1992.

[12] Oliver Aberth. *Precise Numerical Methods Using C++*. Academic Press, 1998.

[13] Nicholas J. Higham† and Theo Mary. A new approach to probabilistic rounding error analysis. *SIAM Journal on Scientific Computing*, 41(5):A2815–A2835, 2019.

[14] B. Liu and T. Kaneko. Error analysis of digital filters realized with floating-point arithmetic. *Proc. IEEE*, 57:p1735–1747, 1969.

[15] B. D. Rao. Floating-point arithmetic and digital filters. *IEEE, Transations on Signal Processing*, 40:85–95, 1992.

[16] Gregory L. Baker and Jerry P. Gollub. *Chaotic Dynamics: An Introduction*. Cambridge University Press, 1990.

[17] Brian Gladman, Vincenzo Innocente, John Mather, and Paul Zimmermann. Accuracy of mathematical functions in single, double, double extended, and quadruple precision. 2024.

[18] R.E. Moore. *Interval Analysis*. Prentice Hall, 1966.

[19] W. Kramer. A prior worst case error bounds for floating-point computations. *IEEE Trans. Computers*, 47:750–756, 1998.

[20] G. Alefeld and G. Mayer. Interval analysis: Theory and applications. *Journal of Computational and Applied Mathematics*, 121:421–464, 2000.

[21] W. Kramer. Generalized intervals and the dependency problem. *Proceedings in Applied Mathematics and Mechanics*, 6:685–686, 2006.

[22] A. Neumaier S.M. Rump S.P. Shary B. Kearfott, M. T. Nakao and P. Van Hentenryck. Standardized notation in interval analysis. *Computational Technologies*, 15:7–13, 2010.

[23] Michael J. Evans and Jeffrey S. Rosenthal. *Probability and Statistics: The Science of Uncertainty*. W. H. Freeman, 2003.

[24] W. T. Tucker and S. Ferson. *Probability bounds analysis in environmental risk assessments*. Applied Biomathmetics, 100 North Country Road, Setauket, New York 11733, 2003.

[25] J. Stolfi and L. H. de Figueiredo. An introduction to affine arithmetic. *TEMA Tend. Mat. Apl. Comput.*, 4:297–312, 2003.

[26] R. Alt and J.-L. Lamotte. Some experiments on the evaluation of functional ranges using a random interval arithmetic. *Mathematics and Computers in Simulation*, 56:17–34, 2001.

[27] J. Stolfi and L. H. de Figueiredo. *Self-validated numerical methods and applications*. ftp:// ftp.tecgraf.puc-rio.br/pub/lhf/doc/cbm97.ps.gz, 1997.

[28] Propagation of uncertainty. `http://en.wikipedia.org/wiki/Propagation_of_uncertainty`, 2011. wikipedia, the free encyclopedia.

[29] S. Ferson H. M. Regan and D. Berleant. Equivalence of methods for uncertainty propagation of real-valued random variables. *International Journal of Approximate Reasoning*, 36:1–30, 2004.

[30] Significance arithmetic. `http://en.wikipedia.org/wiki/Significance_arithmetic`, 2011. wikipedia, the free encyclopedia.

[31] M. Goldstein. Significance arithmetic on a digital computer. *Communications of the ACM*, 6:111–117, 1963.

[32] R. L. Ashenhurst and N. Metropolis. Unnormalized floating-point arithmetic. *Journal of the ACM*, 6:415–428, 1959.

[33] G. Spaletta M. Sofroniou. Precise numerical computation. *The Journal of Logic and Algebraic Programming*, 65:113–134, 2005.

[34] J. Vignes. A stochastic arithmetic for reliable scientific computation. *Mathematics and Computers in Simulation*, 35:233–261, 1993.

[35] C. Denis N. S. Scott, F. Jezequel and J. M. Chesneaux. Numerical 'health' check for scientific codes: the cadna approach. *Computer Physics Communications*, 176(8):501–527, 2007.

[36] J. Hefferon. Linear algebra. `http://joshua.smcvt.edu/linearalgebra/`, 2011.

[37] Paul Horowitz and Hill Winfield. *Art of Electronics*. Cambridge Univ Press, 1995.

[38] Fredrik Gustafsson and Gustaf Hendeby. Some relations between extended and unscented kalman filters. *IEEE Transactions on Signal Processing*, 60-2:545–555, 2012.