DEFINING INTEGER AND FRACTIONAL PARTS CONSISTENTLY WITH POSITIONAL NOTATION

BERTRAND D. THÉBAULT

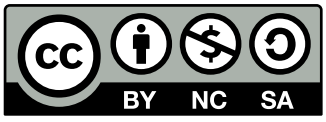
ABSTRACT. Release 1.4: This paper introduces rigorous and consistent definitions for the integer and fractional parts of a real number x within the framework of positional notation. Building on the positional series representation: $x = \mathrm{sgn}(x) \sum_{k=-N}^{n}{}_{N\to\infty} a_k b^k$, I analyze the three main methods:

(1) **Method 1 (Graham, Knuth, & Patashnik, 1992)** [4] satisfies the *Regularity Theorem*, ensuring that the integer part increases by 1 for each increment of x by 1. However, achieving this regularity for negative numbers requires offsets that diverge from positional notation principles, leading to inconsistencies in reconstructing the standard positional representation.
(2) **Method 2a (Daintith, 2004)** [8] is the only method that fully aligns with both the *standard positional number representation* and the *positional series decomposition*, ensuring symmetry for positive and negative values. This approach treats the sign as a global attribute, enabling compliance with IEEE 754 standards and offering a mathematically rigorous, interoperable solution for both theoretical and computational applications.
(3) **Method 2b (Weisstein, MathWorld)** [3] embeds the sign directly into both the integer and fractional parts, simplifying computational implementation. However, this introduces structural inconsistencies for negative numbers, as re-concatenating the parts results in an additional negative sign, diverging from the standard positional representation.

Of these, Method 2a stands out as the most robust approach, combining mathematical rigor, theoretical precision, and practical compatibility. By adopting Method 2a, positional representations of real numbers can achieve consistency, precision, and cross-platform interoperability.

"Only true eternal mathematics is discovered; all other mathematical constructs are at best mere approximations or, at worst, mistaken."

Warning: Average Maturity Index of this final preprint: 9.5/10



Date: 8 January 2025.
2020 *Mathematics Subject Classification.* Primary 03E65.
Key words and phrases. Real number, Integer part, Fractional part, positional notation, IEEE754.

CONTENTS

Part 1. Promoting the Integer and Fractional Parts Consistently with Positional Notation

1. Prerequisites: Definitions of the Floor, Ceiling, and Modulo Functions

Before discussing the central problem of this article, I will review essential definitions on which this work is based. These definitions primarily draw from [1].

Definition 1.1. The floor function $\lfloor x \rfloor$ is the greatest integer less than or equal to x, defined as:

$$\begin{aligned} \lfloor \cdot \rfloor : \mathbb{R} &\to \mathbb{Z} \\ x &\mapsto \lfloor x \rfloor = \max\{k \mid k \leq x,\ k \in \mathbb{Z}\} \end{aligned}$$

Example 1.2. Floor function examples:

- For $x = 3.7$, $\lfloor 3.7 \rfloor = 3$.
- For $x = -2.4$, $\lfloor -2.4 \rfloor = -3$.

Definition 1.3. The ceiling function $\lceil x \rceil$ is the smallest integer greater than or equal to x, defined as:

$$\begin{aligned} \lceil \cdot \rceil : \mathbb{R} &\to \mathbb{Z} \\ x &\mapsto \lceil x \rceil = \min\{k \mid k \geq x,\ k \in \mathbb{Z}\} \end{aligned}$$

Example 1.4. Ceiling function examples:

- For $x = 4.2$, $\lceil 4.2 \rceil = 5$.
- For $x = -1.7$, $\lceil -1.7 \rceil = -1$.

Definition 1.5. The modulo operation $(x \mod m)$ gives the remainder when x is divided by m, defined as:

$$\begin{aligned} (\cdot \mod \cdot) : \mathbb{R} \times \mathbb{N} &\to \mathbb{R} \\ (x, m) &\mapsto x - m \left\lfloor \frac{x}{m} \right\rfloor \end{aligned}$$

Example 1.6. Modulo examples:

- For $x = 17$ and $m = 5$, $\mod 175 = 17 - 5\left\lfloor \frac{17}{5} \right\rfloor = 17 - 15 = 2$.
- For $x = -7$ and $m = 3$, $\mod -73 = -7 - 3\left\lfloor \frac{-7}{3} \right\rfloor = -7 - (-9) = 2$.

2. Real Number Generalized Positional Notation

Definition 2.1. The sign function of a real number x, denoted $\operatorname{sgn}(x)$, is defined as:

$$\forall x \in \mathbb{R} : \quad \operatorname{sgn}(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ -1 & \text{if } x < 0 \end{cases} \tag{2.1}$$

Definition 2.2. A real number x in a positional notation system with base b is expressed as:

$$\forall x \in \mathbb{R} : \quad x = \begin{cases} (a_n a_{n-1} \dots a_0 \cdot a_{-1} a_{-2} \dots)_b & \text{if } x \geq 0 \\ -(a_n a_{n-1} \dots a_0 \cdot a_{-1} a_{-2} \dots)_b & \text{if } x < 0 \end{cases} \iff x = \operatorname{sgn}(x) \sum_{\substack{k=-N \\ N \to \infty}}^{n} a_k b^k \tag{2.2}$$

where:

- b is the base of the number system (e.g., $b = 10$ for the decimal system).
- k is an integer that represents the position of the digit relative to the radix point.
- a_k represents the digits of the number, where $a_k \in \mathbb{Z}(b)$, and $\mathbb{Z}(b) = \{0, 1, 2, \dots, b-1\}$ is the set of possible digits in base b.
- n is the index of the most significant digit a_n and satisfies $n = \lfloor \log_b(x) \rfloor$.
- The integer part of x corresponds to the sum of terms with $k \geq 0$ (positive powers of b), while the fractional part corresponds to the sum with $k < 0$ (negative powers).
- The sign is treated as an external factor; a_k values are always non-negative.

Explicitly, the number is represented as:

$$x = a_n b^n + a_{n-1} b^{n-1} + \cdots + a_0 b^0 + a_{-1} b^{-1} + a_{-2} b^{-2} + \cdots, \tag{2.3}$$

where a_n is the most significant digit with $n = \lfloor \log_b(x) \rfloor$, and $a_{-1}, a_{-2}, \dots$ represent digits in the fractional part.

For negative real numbers $x < 0$, the representation is similar, except the overall positional notation is prefixed by a negative sign:

$$x = -\left(a_n b^n + a_{n-1} b^{n-1} + \cdots + a_0 b^0 + a_{-1} b^{-1} + a_{-2} b^{-2} + \cdots\right). \tag{2.4}$$

When the base b is clear from context, the positional notation may be simplified as:

$$x = \begin{cases} a_n a_{n-1} \dots a_0 \cdot a_{-1} a_{-2} \dots & \text{if } x \geq 0 \\ -a_n a_{n-1} \dots a_0 \cdot a_{-1} a_{-2} \dots & \text{if } x < 0 \end{cases} \iff x = \operatorname{sgn}(x) \sum_{\substack{k=-N \\ N \to \infty}}^{n} a_k b^k \tag{2.5}$$

with $n = \lfloor \log_b(x) \rfloor$.
On the left, **standard positional number representation** uses a comma (or decimal point) as a separator, with the minus sign appearing explicitly before the integer part if $x < 0$.

On the right, **the positional series representation** with the sign function of x, $\operatorname{sgn}(x)$in front of the series and where the $a_k b^k$ terms of the series are non negative $\forall k \in E$ with subset E defined as:

$$E = \{k \in \mathbb{Z} \mid k \leq n = \lfloor \log_b(x) \rfloor\}$$

3. Integer and Fractional Parts Consistent with Positional Notation

Here, I provide a thorough definition of the integer and fractional parts as I originally envisioned them. This is followed by an exploration of other widely used definitions that I discovered later, which are examined and compared in detail in part 2.

Definition 3.1. The integer and fractional parts of $x \in \mathbb{R}$ are defined as follows:

$$x = \begin{cases} \underbrace{a_n a_{n-1} \dots a_0}_{\text{integer part}} \cdot \underbrace{a_{-1} a_{-2} \dots}_{\text{fractional part}} & \text{if } x \geq 0, \\ \underbrace{-a_n a_{n-1} \dots a_0}_{\text{integer part}} \cdot \underbrace{a_{-1} a_{-2} \dots}_{\text{fractional part}} & \text{if } x < 0 \end{cases} \iff x = \underbrace{\operatorname{sgn}(x) \sum_{k=0}^{n} a_k b^k}_{\text{integer part series}} + \operatorname{sgn}(x) \underbrace{\sum_{\substack{k=-N \\ N\to\infty}}^{-1} a_k b^k}_{\text{fraction. part series}} \tag{3.1}$$

where $n = \lfloor \log_b(x) \rfloor$.

On the left, **standard positional number representation** with the minus sign appearing explicitly before the integer part if $x < 0$, **implying the sign applies to both the integer and fractional parts**. When reading from let to right: the integer part's digits appear in front of the comma, while the fractional part's digits appear after the comma.

On the right, **standard positional number representation** is decomposed into the **the integer part series** and **the fractional part series**. Here, the sign function, $\operatorname{sgn}(x)$, is embedded in the integer part (series), while the fractional part (series) remains non-negative.

Definition 3.2. The integer part is equal to **the integer part series** and **the fractional part** is equal to **the fractional part series**: Anticipating part 2's subsection 5.2 describing method 2a , I denote the integer and fractional parts by $\operatorname{int}_2(\cdot)$ and $\operatorname{fract}_{2a}(\cdot)$, respectively, expressed as:

$$\underbrace{\operatorname{int}_2(x)}_{\text{integer part}} = \underbrace{\operatorname{sgn}(x) \sum_{k=0}^{n} a_k b^k}_{\text{integer part series}} \quad \text{and} \quad \underbrace{\operatorname{fract}_{2a}(x)}_{\text{fractional part}} = \underbrace{\sum_{\substack{k=-N \\ N\to\infty}}^{-1} a_k b^k}_{\text{fractional part series}}$$

In this notation, the fractional part $\operatorname{fract}_{2a}(x)$ is always non-negative, since the a_k values are non-negative by definition.

To comply with the standard positional number representation described in equation 2.5 the sign function $\operatorname{sgn}(x)$ must be applied across both the integer part series and the fractional parts series, therefore, the sign function is factored out in the series decomposition, where the integer part includes terms with non-negative powers of b, and the fractional part includes terms with negative powers of b:

$$x = \operatorname{sgn}(x) \left(\sum_{k=0}^{n} a_k b^k + \sum_{\substack{k=-N \\ N\to\infty}}^{-1} a_k b^k \right), \quad \text{where } n = \lfloor \log_b(x) \rfloor. \tag{3.2}$$

4. Expressing the Integer Part Outside Positional Notation

Theorem 4.1. *For a real number x expressed in base b positional series representation, the integer part is given by:*

$$\operatorname{int}_2(x) = \operatorname{sgn}(x) \sum_{k=0}^{\lfloor \log_b(x) \rfloor} a_k b^k = \operatorname{sgn}(x) \cdot \lfloor |x| \rfloor$$

Proof 4.2. To isolate the integer part in the positional notation of x, I start with:

$$\text{int}_2(x) = \text{sgn}(x) \sum_{k=0}^{\lfloor \log_b(x) \rfloor} a_k b^k.$$

To prove that:

$$\text{int}_2(x) = \text{sgn}(x) \cdot \lfloor |x| \rfloor . \text{int}_2(x) = \text{sgn}(x) \lfloor |x| \rfloor,$$

I need to demonstrate:

$$\sum_{k=0}^{\lfloor \log_b(x) \rfloor} a_k b^k = \lfloor |x| \rfloor.$$

For $x \geq 0$:

$$\text{int}_2(x) = \sum_{k=0}^{\lfloor \log_b(x) \rfloor} a_k b^k = \lfloor x \rfloor = \lfloor |x| \rfloor.$$

For $x < 0$:

$$\text{int}_2(x) = - \sum_{k=0}^{\lfloor \log_b(x) \rfloor} a_k b^k = - \lfloor |x| \rfloor.$$

By eliminating the negative sign in the final terms, I have:

$$\sum_{k=0}^{\lfloor \log_b(x) \rfloor} a_k b^k = \lfloor |x| \rfloor.$$

This establishes that for $x \in \mathbb{R}$:

$$\text{int}_2(x) = \text{sgn}(x) \lfloor |x| \rfloor.$$

Part 2. **Analysing Conflicting Definitions of Integer and Fractional Parts**

5. Conflicting Definitions for Integer and Fractional Parts

For negative real numbers, sources such as the English version of Wikipedia [2] and MathWorld [3] have highlighted multiple approaches with conflicting definitions for the integer and fractional parts. Specifically, two conflicting definitions exist for the integer part and three for the fractional part. Each of these definitions is described, analyzed, and compared in the following sections.

5.1. **Method 1: Regular for Unspecified Notation System (Graham, Knuth & Patashnik 1992, YDNGWYS)**

5.1.1. *Definitions* The integer and fractional parts of $x \in \mathbb{R}$ are defined as follows:

Integer Part:

$$\text{int}_1(x) = \lfloor x \rfloor = \text{sgn}(x) \cdot \sum_{k=0}^{n} a_k b^k - \frac{1}{2}(1 - \text{sgn}(x)), \quad \text{where} \quad n = \lfloor \log_b(x) \rfloor$$

where $n = \lfloor \log_b(|x|) \rfloor$, and:

$$\frac{1}{2}(\text{sgn}(x) - 1) = \begin{cases} 0 & \text{if } x \geq 0, \\ 1 & \text{if } x < 0. \end{cases}$$

Fractional Part:

$$\text{fract}_1(x) = x - \lfloor x \rfloor = \sum_{k=1}^{n} a_{-k} \cdot b^{-k} + \frac{1}{2}(1 - \text{sgn}(x)).$$

Overall Formula:

$$x = \mathrm{int}_1(x) + \mathrm{fract}_1(x) = \mathrm{sgn}(x) \cdot \sum_{\substack{k=-N \\ N\to\infty}}^{n} a_k b^k.$$

5.1.2. *Regularity and Theorem Proof* This method is termed **regular** on the integer part because it satisfies the following relationships:

$$\mathrm{fract}_1(x) \in [0, 1) \quad \text{and} \quad \mathrm{int}_1(x+1) - \mathrm{int}_1(x) = 1 \quad \forall x \in \mathbb{R}.$$

The **Regularity Theorem** states:

$$\forall x \in \mathbb{R}, \quad \mathrm{int}_1(x+1) - \mathrm{int}_1(x) = 1.$$

Proof:

$$\mathrm{int}_1(x+1) = \lfloor x+1 \rfloor = \lfloor x \rfloor + 1, \quad \text{since } x + 1 - \lfloor x+1 \rfloor = x - \lfloor x \rfloor.$$

Thus:

$$\mathrm{int}_1(x+1) - \mathrm{int}_1(x) = \lfloor x+1 \rfloor - \lfloor x \rfloor = 1.$$

5.1.3. *Positional Regularity Analysis* For positive x, both $\mathrm{int}_1(x)$ and $\mathrm{fract}_1(x)$ align with the standard positional notation series. However, for $x < 0$, the fractional part deviates from the series $\sum_{\substack{k=-N \\ N\to\infty}}^{-1} a_k b^k$, as:

$$\mathrm{fract}_1(x) = x - \lfloor x \rfloor > 0 \quad \text{(always non-negative)}.$$

This discrepancy arises because:

(1) The integer part $\mathrm{int}_1(x)$ introduces an offset by -1 for $x < 0$, since the flooring operation $\lfloor x \rfloor$ corresponds to:

$$\lfloor x \rfloor = \mathrm{sgn}(x) \cdot \sum_{k=0}^{n} a_k b^k - \frac{1}{2}(1 - \mathrm{sgn}(x)).$$

Specifically, the subtraction of $\frac{1}{2}(1 - \mathrm{sgn}(x))$ for negative x ensures that $\lfloor x \rfloor$ "floors" x to the nearest smaller integer.

(2) The fractional part $\mathrm{fract}_1(x)$ compensates for this offset by introducing a term $\frac{1}{2}(1 - \mathrm{sgn}(x))$, leading to:

$$\mathrm{fract}_1(x) = \sum_{k=1}^{n} a_{-k} \cdot b^{-k} + \frac{1}{2}(1 - \mathrm{sgn}(x)).$$

This artificial compensation satisfies the regularity theorem but prevents re-concatenation of the integer and fractional parts to form the standard positional number representation as defined in equation 2.5.

5.1.4. *Example Analysis* Consider $x = -0.3$:

- **Integer part**:

$$\mathrm{int}_1(-0.3) = \lfloor -0.3 \rfloor = -1.$$

Using the formula:

$$\mathrm{int}_1(-0.3) = -1 \cdot 0 - \frac{1}{2}(1 - (-1)) = -1.$$

- **Fractional part**:

$$\mathrm{fract}_1(-0.3) = -0.3 - \lfloor -0.3 \rfloor = -0.3 - (-1) = 0.7.$$

Using the formula:

$$\mathrm{fract}_1(-0.3) = \sum_{k=1}^{n} a_{-k} \cdot b^{-k} + \frac{1}{2}(1 - (-1)) = 0 + 0.7 = 0.7.$$

- **Result**:

$$x = \text{int}_1(x) + \text{fract}_1(x) = -1 + 0.7 = -0.3.$$

5.1.5. *Conclusion: Why YDNGWYS?* This widely used regular method fails to align with positional notation for negative x, due to:

(1) **Positional Misalignment**: The fractional part does not represent the fractional part series for $x < 0$.
(2) **Artificial Offsets**: The components $\text{int}_1(x)$ and $\text{fract}_1(x)$ include offsets to satisfy the regularity theorem but prevent re-concatenation into the standard positional number representation.

Thus, this method is aptly described as **"You Do Not Get What You See" (YDNGWYS)**, as it fails to provide a straightforward, positional interpretation of the components for all $x < 0$.

5.2. Method 2a: Odd Function with Positive Fractional Part (WYSIWYG)

5.2.1. *Core Definitions:*

(1) **Integer Part:**

$$\text{int}_2(x) = \text{sgn}(x) \cdot \lfloor |x| \rfloor = \text{sgn}(x) \cdot \sum_{k=0}^{n} a_k \cdot b^k, \quad \text{where} \quad n = \lfloor \log_b(x) \rfloor$$

This aligns with the integer part series described in part 1's equation **Equation 3.1**, ensuring compatibility with the standard positional notation system. The integer part satisfies the **odd function property**:

$$\text{int}_2(-x) = -\text{int}_2(x).$$

(2) **Fractional Part (Daintith, 2004):**

$$\text{fract}_{2a}(x) = |x| - \lfloor |x| \rfloor = \sum_{\substack{k=-N \\ N\to\infty}}^{-1} a_k \cdot b^k.$$

This definition ensures $\text{fract}_{2a}(x) \geq 0$, aligning with the fractional part series described in **Equation 3.1**, preserving strict compliance with the positional notation decomposition.

(3) **Overall Formula:**

$$x = \text{int}_2(x) + \text{sgn}(x) \cdot \text{fract}_{2a}(x) = \text{sgn}(x) \cdot \sum_{\substack{k=-N \\ N\to\infty}}^{n} a_k \cdot b^k.$$

This representation explicitly adheres to **Equation 3.2** in part 1, where the sign function $\text{sgn}(x)$ applies uniformly across both the integer and fractional part series.

Theorem 5.1. *Method 2a's overall formula is an odd function, i.e., for all* $x \in \mathbb{R}$,

$$int_2(-x) + sgn(-x) \cdot fract_{2a}(-x) = -\left(int_2(x) + sgn(x) \cdot fract_{2a}(x)\right) = -x$$

Proof 5.2. I show that both integer and fractional parts in Method 2a satisfy the odd function property:

(1) **Integer Part:**
By definition in Method 2a,

$$\text{int}_2(x) = \text{sgn}(x) \cdot \lfloor |x| \rfloor.$$

$\text{int}_2(x)$ is an odd function:

$$\text{int}_2(-x) = \text{sgn}(-x) \cdot \lfloor |-x| \rfloor = -\text{sgn}(x) \cdot \lfloor |x| \rfloor = -\text{int}_2(x).$$

(2) **Fractional Part:**
The fractional part is defined as:

$$\text{fract}_{2a}(x) = |x| - \lfloor |x| \rfloor.$$

For $-x$,

$$\text{fract}_{2a}(-x) = |-x| - \lfloor |-x| \rfloor = |x| - \lfloor |x| \rfloor = \text{fract}_{2a}(x).$$

So, $\text{fract}_{2a}(x)$ is an even function.

Combining integer and fractional parts for $-x$:

$$\text{int}_2(-x) + \text{sgn}(-x) \cdot \text{fract}_{2a}(-x) = -\text{int}_2(x) - \text{sgn}(x) \cdot \text{fract}_{2a}(x) = -\left(\text{int}_2(x) + \text{sgn}(x) \cdot \text{fract}_{2a}(x)\right).$$

This confirms that the overall expression used in Method 2a is an odd function. This ensures symmetry of the decomposition with respect to the origin, consistent with positional notation principles.

5.2.2. *Example Analysis:* For $x = -0.3$:

(1) **Absolute Value:** $|x| = 0.3$.
(2) **Integer Part:**

$$\text{int}_2(-0.3) = \text{sgn}(-0.3) \cdot \left\lfloor \left|-0.3\right| \right\rfloor = -1 \cdot \lfloor 0.3 \rfloor = -1 \cdot 0 = 0.$$

(3) **Fractional Part:**

$$\text{fract}_{2a}(-0.3) = \left|-0.3\right| - \left\lfloor \left|-0.3\right| \right\rfloor = 0.3 - 0 = 0.3.$$

(4) **Combined Result:**

$$x = \text{int}_2(-0.3) + \text{sgn}(-0.3) \cdot \text{fract}_{2a}(-0.3) = 0 + (-1) \cdot 0.3 = -0.3.$$

5.2.3. *Remarks:*

(1) **Strict Compliance with Positional Notation:** - Method 2a is the only method that complies with the **standard positional number representation** (**Equation 2.5**) for both positive and negative numbers, preserving the integrity of the decomposition.
(2) **Integer Part Behavior:** - The integer part behaves as an **odd function**:

$$\text{int}_2(-x) = -\text{int}_2(x).$$

- For a value like -0.3, the integer part is -0. Here, 0 as an integer part can be either positive or negative. However, this ambiguity can be avoided, as the sign function $\text{sgn}(x)$ is only embedded in the integer part, it can be factored out to comply with equation 3.2
(3) **Fractional Part Consistency:** - The fractional part is always non-negative, aligning with the requirements of positional notation. Its value directly corresponds to the positional series for negative powers of b, ensuring strict compliance with **Equations 3.1 and 3.2**.
(4) **Compatibility with IEEE 754:** - by allowing a single sign function to be factored out from the series decomposition, Method 2a naturally aligns with **IEEE 754 floating-point standards** as described in Section 5.5), where the sign bit ensures a clear distinction between $+0$ and -0. This compatibility is a practical advantage in computational contexts.

5.2.4. *Conclusion:*

- Method 2a is the **only method** that fully respects both the standard positional number representation and the positional series decomposition, ensuring the sign applies uniformly across both integer and fractional parts for negative values. Thus the method allow the integer part and the fractional part to be re-concatenated into the standard positional number representation, deserving the calling WYIWYG: What You See Is What You Get.
- While primarily grounded in positional notation principles, the method's clear handling of the sign function ensures computational compatibility with **IEEE 754**.
- This method provides a rigorous and interpretable framework for representing real numbers, offering theoretical clarity and practical computational reliability.

5.3. Method 2b: Odd Function YDNGWYS with Signed Fractional Part (Wolfram Mathematica)

5.3.1. *Definitions:*

- **Integer Part:**

$$\mathrm{int}_2(x) = \mathrm{sgn}(x) \cdot \lfloor |x| \rfloor = \mathrm{sgn}(x) \cdot \sum_{k=0}^{n} a_k \cdot b^k \quad \text{where} \quad n = \lfloor \log_b(x) \rfloor$$

- **Signed Fractional Part:**

$$\mathrm{fract}_{2b}(x) = \mathrm{sgn}(x) \cdot \big(|x| - \lfloor |x| \rfloor \big) = \mathrm{sgn}(x) \cdot \sum_{\substack{k=-N \\ N\to\infty}}^{-1} a_k \cdot b^k.$$

- **Overall Formula:**

$$x = \mathrm{int}_2(x) + \mathrm{fract}_{2b}(x) = \mathrm{sgn}(x) \cdot \sum_{\substack{k=-N \\ N\to\infty}}^{n} a_k \cdot b^k.$$

5.3.2. *Example Analysis:* For $x = -0.3$:

- **Integer part:**

$$\mathrm{int}_2(-0.3) = \mathrm{sgn}(-0.3) \cdot \lfloor 0.3 \rfloor = -1 \cdot 0 = 0.$$

- **Signed fractional part:**

$$\mathrm{fract}_{2b}(-0.3) = \mathrm{sgn}(-0.3) \cdot \big(0.3 - 0 \big) = -0.3.$$

- **Combined result:**

$$x = \mathrm{int}_2(-0.3) + \mathrm{fract}_{2b}(-0.3) = 0 + (-0.3) = -0.3.$$

5.3.3. *Remarks:*

- **Positional Compliance:** - Method 2b is compliant with the positional series decomposition for positive and negative numbers:

$$\forall x \in \mathbb{R} : \quad x = \mathrm{sgn}(x) \sum_{\substack{k=-N \\ N\to\infty}}^{n} a_k b^k = \underbrace{\mathrm{sgn}(x) \sum_{k=0}^{n} a_k b^k}_{\text{integer part series}} + \underbrace{\mathrm{sgn}(x) \sum_{\substack{k=-N \\ N\to\infty}}^{-1} a_k b^k}_{\text{fractional part series}} \tag{5.1}$$

where $n = \lfloor \log_b(x) \rfloor$.

- However, for negative x Method 2b does not comply with the standard positional number representation for negative number as the re-concatenation of the integer part with the fractional part is going to lead to an extra negative sign:

$$x = \begin{cases} \underbrace{a_n a_{n-1} \dots a_0}_{\text{integer part}} \cdot \underbrace{a_{-1} a_{-2} \dots}_{\text{fractional part}} & \text{if } x \geq 0, \\ \underbrace{-a_n a_{n-1} \dots a_0}_{\text{integer part}} \cdot \underbrace{-a_{-1} a_{-2} \dots}_{\text{fractional part}} & \text{if } x < 0 \end{cases} \iff x = \underbrace{\mathrm{sgn}(x) \sum_{k=0}^{n} a_k b^k}_{\text{integer part series}} + \underbrace{\mathrm{sgn}(x) \sum_{\substack{k=-N \\ N\to\infty}}^{-1} a_k b^k}_{\text{fractional part series}} \tag{5.2}$$

where $n = \lfloor \log_b(x) \rfloor$. So Method 2a fails and also become YDNGWYS (You Do Not Get What You See).

- **Sign Embedding:** - This approach embeds the sign directly into both the integer and fractional parts, eliminating the need for a separate $\mathrm{sgn}(x)$ component. However, this introduces structural differences from the standard positional representation for negative values.

- **Implementation Considerations:** - The design prioritizes computational simplicity by avoiding external sign management, a choice likely influenced by Wolfram Mathematica's conventions.

5.3.4. *Conclusion:*

(1) Method 2b respect only comply with positional series decomposition, but does not respects the standard positional number representation for negative number because of the extra negative sign for the re-concatenation of the integer part and the fractional part
(2) however it provides a consistent framework for computer oriented optimisation requiring signed fractional representation, or avoidance of external sign structure. However sign embedding introduces potential challenges, such as discrepancies due to rounding or truncation errors

5.4. **IEEE 754 Standard: General Overview** The IEEE 754 standard [9], first established in 1985, introduced a widely adopted approach to representing floating-point numbers in computing. This standard was developed to provide a consistent method for handling real numbers across different hardware and software, ensuring compatibility and predictability in numerical calculations. The IEEE 754 format presents numbers with three main components: the sign, the exponent, and the mantissa (or significand).

In the IEEE 754 binary floating-point format, the sign bit determines the overall sign of the number. The exponent, which is biased, adjusts the scale of the number, while the mantissa represents the precision. Floating-point numbers are expressed as:

$$\text{sign} \times \text{mantissa} \times 2^{\text{exponent}}.$$

For a single-precision floating-point number, 32 bits are used, with 1 bit for the sign, 8 bits for the exponent, and 23 bits for the mantissa. In double precision, 64 bits are allocated: 1 for the sign, 11 for the exponent, and 52 for the mantissa. This structure allows a wide dynamic range but introduces challenges in maintaining precision across all real numbers.

5.4.1. *Widespread Adoption and Limitations* The IEEE 754 standard has been universally adopted in almost all modern computing systems due to its standardized representation, which supports both performance and compatibility across platforms. This standardization facilitates numerical computations in diverse domains such as scientific simulations, machine learning, and real-time systems.

However, one inherent limitation of IEEE 754 is its approximation of real numbers. Certain values cannot be represented exactly due to finite bit constraints, leading to minor precision errors. These errors can accumulate in iterative calculations, particularly when dealing with very small or very large numbers. Nonetheless, these limitations reflect the finite nature of hardware representation rather than flaws in the IEEE 754 design itself.

5.5. **IEEE 754 Bias Representation and Its Mathematical Basis** In the IEEE 754 floating-point standard [9], real numbers are represented in the following format:

$$x = (-1)^s \cdot (1 + M) \cdot 2^e,$$

where:

- s is the sign bit ($s = 0$ for positive numbers, $s = 1$ for negative numbers),
- M is the mantissa, a normalized fraction in $[0, 1)$,
- e is the actual exponent, determining the scale of the number.

To encode e into an n-bit field E_b for storage in binary, the standard applies a **bias**, defined as:

$$\text{bias} = 2^{n-1} - 1.$$

The bias shifts e into a non-negative range, allowing unsigned integer storage:

$$E_b = e + \text{bias}.$$

To retrieve the actual exponent:

$$e = E_b - \text{bias}.$$

5.5.1. *Range of Exponent* For an n-bit exponent field:

$$e_{\min} = 1 - \text{bias} = 1 - (2^{n-1} - 1) = -(2^{n-1} - 2),$$
$$e_{\max} = 2^n - 2 - \text{bias} = (2^n - 2) - (2^{n-1} - 1) = 2^{n-1} - 1.$$

For example:

- Single precision ($n = 8$, bias $= 127$): $e \in [-126, 127]$,
- Double precision ($n = 11$, bias $= 1023$): $e \in [-1022, 1023]$.

5.5.2. *Encoded Representation and Special Cases* Given the representation:

$$x = (-1)^s \cdot (1 + M) \cdot 2^{E_b - \text{bias}},$$

special cases are defined as follows:

- $E_b = 0$: Represents denormalized numbers with $e = 1 - \text{bias}$.
- $E_b = 2^n - 1$: Reserved for $+\infty, -\infty$ (when $M = 0$) or NaN (when $M \neq 0$).
- $E_b = \text{bias}$: Represents numbers with $e = 0$, allowing representation of both $+0$ and -0.

5.6. **Compliance with Positional Representation** The IEEE 754 bias-based exponent encoding aligns with the principles of positional notation. The explicit separation of the sign bit and the biased exponent ensures:

- A consistent method for encoding both positive and negative exponents,
- Compatibility with Method 2a (see Section 5.2), where the sign is treated as a global attribute and the fractional part remains positive.

This alignment demonstrates the rigor of IEEE 754 in maintaining consistency with mathematical principles while optimizing for hardware implementation.

Example 5.3. For a single-precision floating-point number with:

- Sign bit $s = 0$,
- Exponent bits $E_b = 130$ (10000010_2),
- Mantissa $M = 0.5$,

I compute:

$$e = E_b - \text{bias} = 130 - 127 = 3,$$

and:

$$x = (-1)^0 \cdot (1 + 0.5) \cdot 2^3 = 1.5 \cdot 8 = 12.$$

This example illustrates how IEEE 754 enables consistent encoding and decoding of real numbers, leveraging bias to ensure a wide dynamic range.

6. Conclusion

This paper analyzed three competing methods for defining the integer and fractional parts of real numbers, with an emphasis on their compliance with positional notation and their mathematical properties:

(1) **Method 1: Regular YDNGWYS for Unspecified Notation System (Graham, Knuth, & Patashnik, 1992)** [4]: Method 1 satisfies the *Regularity Theorem*, ensuring that:

$$\text{int}_1(x + 1) - \text{int}_1(x) = 1 \quad \forall x \in \mathbb{R}.$$

However, achieving this regularity for negative numbers requires an offset in the integer and fractional parts. This offset ensures that the fractional part of $x < 0$ is non-negative, but it introduces a deviation from the standard positional series decomposition. Consequently,

Method 1 cannot re-concatenate its integer and fractional parts to reconstruct the standard positional number representation, leading to inconsistencies in negative number representation. This trade-off renders it a *You Do Not Get What You See (YDNGWYS)* approach, as its components do not align with positional notation principles.

(2) **Method 2a: Odd Function WYSIWYG with Positive Fractional Part (Daintith, 2004)** [8]: Method 2a achieves perfect compliance with both the *standard positional number representation* and the *positional series decomposition*. By treating the sign as a global attribute and ensuring that the fractional part is always non-negative, Method 2a provides a mathematically rigorous and symmetric representation for both positive and negative numbers. It also adheres to the IEEE 754 floating-point standard by factoring the sign out as a single attribute. This makes it a true *What You See Is What You Get (WYSIWYG)* approach, combining theoretical rigor with practical compatibility for computational applications.

(3) **Method 2b: Odd Function YDNGWYS with Signed Fractional Part (Weisstein, MathWorld)** [3]: Method 2b embeds the sign into both the integer and fractional parts, eliminating the need for an external sign attribute. While this design choice simplifies certain computational tasks, it diverges from the *standard positional number representation* for negative numbers. Specifically, re-concatenating the integer and fractional parts introduces an extra negative sign. This structural inconsistency, while manageable in some contexts, limits its theoretical rigor. Like Method 1, Method 2b can also be characterized as a *You Do Not Get What You See (YDNGWYS)* approach for negative numbers.

Key Insights and Conclusion:

- While Method 1 is regular and widely used, its reliance on offsets to ensure the non-negativity of the fractional part for negative numbers limits its alignment with positional notation.
- Method 2b offers computational simplicity but sacrifices consistency with the standard positional number representation for negative numbers.
- Method 2a emerges as the optimal choice, achieving perfect compliance with positional notation principles while preserving mathematical rigor and cross-platform compatibility. Its alignment with IEEE 754 standards and its ability to re-concatenate the integer and fractional parts into a standard positional representation make it the most robust approach for both theoretical and computational applications.

By adopting Method 2a, positional representations of real numbers can achieve greater interoperability and precision across a wide range of mathematical and computational domains.

REFERENCES

[1] Graham, R. L., Knuth, D. E., & Patashnik, O. (1992).
page x: A Note on Notation:
$\lfloor x \rfloor$ floor: $\max\{n \mid \text{integer } n, n \le x\}$ p: 67
$\lceil x \rceil$ ceil: $\min\{n \mid \text{integer } n, n \ge x\}$ p: 67
mod xy remainder: $\bmod xy = x - \lfloor x/y \rfloor$ p: 82

[2] Wikipedia contributors. (n.d.). Integer part and fractional part. In *Wikipedia, The Free Encyclopedia*. Retrieved from https://en.wikipedia.org/wiki/Integer_part_and_fractional_part

[3] Weisstein, E. W. (n.d.). Integer Part and Fractional Part. In *MathWorld – A Wolfram Web Resource*. Retrieved from http://mathworld.wolfram.com/IntegerPart.html

[4] Graham, R. L., Knuth, D. E., & Patashnik, O. (1992). *Concrete Mathematics: A Foundation for Computer Science*. Addison-Wesley.

[5] Weisstein, Eric W. "Integer Part." Wolfram Research. (n.d.). *From MathWorld–A Wolfram Web Resource.* Retrieved from https://mathworld.wolfram.com/IntegerPart.html (Accessed: 2024-08-31).

[6] Spanier, J., & Oldham, K. B. (1987). *The Integer-Value Int(x) and Fractional-Value frac(x) Functions*. In *An Atlas of Functions* (pp. 71-78). Washington, DC: Hemisphere Publishing.

[7] B. E. S. K. M. M. A. P. S. (2013). *Mathematical Software ICMS 2014*. Springer.

[8] Daintith, J. (2004). *A Dictionary of Computing*. Oxford University Press. ISBN 9780198608776. LCCN 2004276619. Series: Oxford Paperbacks. Available at: https://books.google.fr/books?id=Hay6vTsGFAsC.
[9] IEEE Computer Society. *IEEE Standard for Floating-Point Arithmetic*, IEEE Std 754-2019, 2019.

Thanks to Lé Nguyen Hoang with his Youtube Channel Science4all, who made me discover "La diagonale dévastatrice de Cantor - Infini 16" which motivated me to restart Mathematics at age 55...

Thanks to my family for standing by me, sharing countless evenings and weekends, and supporting me throughout this personal journey of home-based research.

BitCliff Ltd c/o Bertrand THEBAULT 27, Old Gloucester Street London WC1N 3AX United Kingdom
Email address: bertrand@boldrift.com