

Floating-Points Are as Easy as 01, 10, 11!

Why Your Numbers May Be Lying to You...

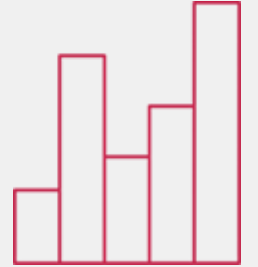
PHUSE EU CONNECT: November 2025



Numeric Representation and Precision

What is numeric representation?

- The format or system in which we express numbers.
 - E.g., the fraction $\frac{1}{10}$ has a decimal representation of 0.1.



What is numeric precision?

- The level of detail we present a number to.



Floating-Point Representation

01

What is Floating-Point Representation?

Floating-point representation is how SAS, and most other programming languages, chooses to represent real numbers.



02

What Floating-Point Representation Does SAS Use?

SAS uses IEEE 754 double-precision floating-point representation.



03

Limitations of Floating-Point Representation

It cannot exactly store all numbers, leading to small inaccuracies.



What is a Binary Number?

Concept

Binary: Binary is a number system that only uses two digits, 0 and 1.

Base-2 System: Binary is called a base-2 numeral system.

How It Works: Each position in a binary number represents a power of 2, starting from the right.

1

The binary number 11 means:

$2 (2^1)$	$1 (2^0)$
1	1

$$\text{Total} = (1 \times 2) + (1 \times 1) = 2 + 1 = 3$$

2

The binary number 0.101 means:

$\frac{1}{2} (2^{-1})$	$\frac{1}{4} (2^{-2})$	$\frac{1}{8} (2^{-3})$
1	0	1

$$\text{Total} = (1 \times \frac{1}{2}) + (0 \times \frac{1}{4}) + (1 \times \frac{1}{8}) = 0.5 + 0 + 0.125 = 0.625$$

Floating-Point Errors

01

What are Floating-Point Errors?

Small inaccuracies in numerical calculations due to the way numbers are stored and processed in our computer's memory.

02

Infinitely repeating representations

Like decimal representation, there are some binary values that have infinitely repeating representations:

- Decimal : $\frac{2}{3} = 0.66666\dots$ with the 6s repeated indefinitely.
- Binary : $\frac{1}{10} = 0.000110011001100110011\dots$ where the pattern 0011 is repeated indefinitely.

03

An infinitely repeating representation in binary is therefore rounded or truncated when stored on a computer.

Floating-Point Errors Continued

- Imprecision can grow as a result of repeatedly performing calculations on truncated/rounded numbers.
- Example using decimal representation:
 - $\frac{2}{3}$ rounded to 3dp = 0.667
 - As a result of performing repeated calculations, we get:
 - $\frac{2}{3} + \frac{2}{3} + \frac{2}{3} = 0.667 + 0.667 + 0.667 = \mathbf{2.01}$
- Same issue with infinitely repeating binary values.

How Floating-Point Errors Can Affect us

01

IF, SELECT and WHERE statements may not produce desired results when using calculated values that may have been rounded or truncated.



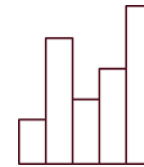
02

Using PROC COMPARE on SAS datasets might show differences in variables that look as though they should be equal.



03

Statistical models may behave unpredictably when comparing numeric values with floating-point rounding involved.



Floating-Point Error in Action

```
data PhuseEU2025;
```

```
*Create a computed variable which adds 10 amounts of 1/10;
```

```
computed= (1/10) + (1/10) + (1/10) + (1/10) + (1/10) +  
           1         2         3         4         5  
           (1/10) + (1/10) + (1/10) + (1/10) + (1/10);  
           6         7         8         9        10
```

```
*Create a specified variable which assigns value 1;
```

```
specified= 1;
```

```
*Compare computed variable and specified variable and flag if equal;
```

```
if computed = specified then help = "Y";
```

```
run;
```

Total rows: 1 Total columns: 3

COMPUTED	
1	1

Drum Roll ...



Floating-Point Error in Action

```
data PhuseEU2025;
```

```
*Create a computed variable which adds 10 amounts of 1/10;
```

```
computed= (1/10) + (1/10) + (1/10) + (1/10) + (1/10) +  
           1         2         3         4         5  
           (1/10) + (1/10) + (1/10) + (1/10) + (1/10);  
           6         7         8         9        10
```

```
*Create a specified variable which assigns value 1;
```

```
specified= 1;
```

```
*Compare computed variable and specified variable and flag if equal;
```

```
if computed = specified then help = "Y";
```

```
run;
```

Total rows: 1 Total columns: 3

	COMPUTED	SPECIFIED	HELP
1	1	1	...



Floating-Point Error in Action

```
data PhuseEU2025;
```

```
*Create a computed variable which adds 10 amounts of 1/10;
```

```
computed= (1/10) + (1/10) + (1/10) + (1/10) + (1/10) +  
           1         2         3         4         5  
           (1/10) + (1/10) + (1/10) + (1/10) + (1/10);  
           6         7         8         9        10
```

```
*Create a specified variable which assigns value 1;
```

```
specified= 1;
```

```
*Compare computed variable and specified variable and flag if equal;
```

```
if computed = specified then help = "Y";
```

```
run;
```



Total rows: 1 Total columns: 3

	COMPUTED	SPECIFIED	HELP
1	1	1	

How to Handle Floating-Point Errors

Round Function

- We can use the ROUND function to ensure that the computed numeric values are represented consistently, with as much precision as applicable.

Proc Compare Criterion Option

- Sets the tolerance for numeric comparisons, controls how SAS decides whether two numeric values are "equal," even if they're not exactly equal due to floating-point error.

```
proc compare base=base compare=compare criterion=0.0001 method = absolute listall error warning;  
run;
```

How to Handle Floating-Point Errors Continued

```
data base;  
input id value;  
datalines;  
1 10.0000  
2 20.0000  
3 30.0000  
;  
run;
```

Total rows: 3 Total columns: 2

	ID	VALUE
1	1	10
2	2	20
3	3	30

```
data compare;  
input id value;  
datalines;  
1 10.0001  
2 20.0002  
3 30.0000  
;  
run;
```

Total rows: 3 Total columns: 2

	ID	VALUE
1	1	10.0001
2	2	20.0002
3	3	30

1. ID Var 1 will have a difference of **0.0001**
2. ID Var 2 will have a difference of **0.0002**
3. ID Var 3 has **no difference**

How to Handle Floating-Point Errors Continued

Total rows: 3 Total columns: 2

	ID	VALUE
1	1	10
2	2	20
3	3	30

Total rows: 3 Total columns: 2

	ID	VALUE
1	1	10.0001
2	2	20.0002
3	3	30

```
proc compare base=base compare=compare  
listall error warning;  
run;
```

The COMPARE Procedure
Comparison of WORK.BASE with WORK.COMPARE
(Method=EXACT)

Value Comparison Results for Variables

Obs		Base VALUE	Compare VALUE	Diff.	% Diff
1		10.0000	10.0001	0.0001000	0.001000
2		20.0000	20.0002	0.000200	0.001000

How to Handle Floating-Point Errors Continued

Total rows: 3 Total columns: 2

	ID	VALUE
1	1	10
2	2	20
3	3	30

Total rows: 3 Total columns: 2

	ID	VALUE
1	1	10.0001
2	2	20.0002
3	3	30

```
proc compare base=base compare=compare  
  criterion=0.0001 method = absolute  
  list all error warning;  
run;
```

The COMPARE Procedure
Comparison of WORK.BASE with WORK.COMPARE
(Method=ABSOLUTE, Criterion=0.0001)

Value Comparison Results for Variables

Obs		Base VALUE	Compare VALUE	Diff.	% Diff
2		20.0000	20.0002	0.000200	0.001000

Conclusion

Floating-point errors are a byproduct of how computers store numbers. They're not bugs—but you need to understand them, especially as a SAS programmer in clinical research, where calculations and manipulation of data is crucial.

Helpful Resources

<https://pharmasug.org/proceedings/2014/CC/PharmaSUG-2014-CC50.pdf>



<https://support.sas.com/resources/papers/proceedings11/275-2011.pdf>



https://documentation.sas.com/doc/en/pgmsascdc/9.4_3.5/lepg/p0dv87zb3bnse6n1mqo360be70qr.htm



Helpful Resources

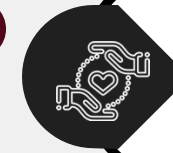
<https://www.youtube.com/watch?v=PZRI1lfStY0>



<https://library.fiveable.me/numerical-analysis-i/unit-2>



https://en.wikipedia.org/wiki/List_of_types_of_numbers



Thank You!

*Why did the binary number break up with the decimal number?
Because it couldn't deal with all the extra digits!*

Chloe Floodgate

Chloe.floodgate@phastar.com

