# Part 7: Comparative Study — IEEE 754 Handling in Python vs. JavaScript

## Overview

Both Python and JavaScript use IEEE 754 double-precision (64-bit) floating-point arithmetic by default, but they differ in their behavior, control over rounding, and error handling.

## 1. Default Precision

| Feature | Python | JavaScript |
|---|---|---|
| Number type | float = IEEE 754 64-bit | Number = IEEE 754 64-bit |
| Decimal support | Optional via decimal module | No built-in high-precision decimal |
| Example | 1.1 + 2.2 = 3.3000000000000003 | Same in JS |

## 2. Rounding Behavior

- Python offers full control using the decimal module with modes like ROUND_HALF_UP, ROUND_DOWN.
- JavaScript uses binary floating-point math with limited rounding control.

### Python Example:

```
from decimal import Decimal, ROUND_HALF_UP
Decimal('1.235').quantize(Decimal('1.00'), rounding=ROUND_HALF_UP)  # 1.24
```

### JavaScript Example:

```
Math.round(1.235 * 100) / 100;  // May return 1.24, not always reliable
```

## 3. Special Values

Both languages support:

- Infinity

- -Infinity
- NaN (Not a Number)

### Python:

float('inf'), float('-inf'), float('nan')

### JavaScript:

Infinity, -Infinity, NaN

Note: In both, NaN != NaN (evaluates to True).

## 4. Underflow & Overflow

| Case | Python Output | JavaScript Output |
|---|---|---|
| Underflow | 0.0 | 0 |
| Overflow | inf | Infinity |

## 5. Denormalized Numbers

- Both support subnormal values, though behavior varies.
- Python: supports values like 5e-324
- JavaScript: depends on browser/engine behavior

## 6. Observed Differences

| Behavior | Python | JavaScript |
|---|---|---|
| Arbitrary precision | Yes (with decimal) | No (use libraries like Big.js) |
| Rounding control | Full via decimal | Limited |
| NaN comparison | nan != nan is True | NaN !== NaN is True |
| Visualization tools | Easy with Matplotlib | Needs browser-based plotting |

## Conclusion

While both languages conform to IEEE 754, Python provides much more visibility and control over floating-point operations. JavaScript, while consistent with IEEE 754, is limited in precision control and visualization without external libraries.