

# Package ‘fpCompare’

July 22, 2025

**Type** Package

**Title** Reliable Comparison of Floating Point Numbers

**URL** <https://github.com/PredictiveEcology/fpCompare>

**Version** 0.2.4

**Date** 2022-08-13

**Description** Comparisons of floating point numbers are problematic due to errors associated with the binary representation of decimal numbers.

Despite being aware of these problems, people still use numerical methods that fail to account for these and other rounding errors (this pitfall is the first to be highlighted in Circle 1 of Burns (2012)

'The R Inferno' <[https://www.burns-stat.com/pages/Tutor/R\\_inferno.pdf](https://www.burns-stat.com/pages/Tutor/R_inferno.pdf)>).

This package provides new relational operators useful for performing floating point number comparisons with a set tolerance.

**Depends** R (>= 3.4)

**Suggests** covr, knitr, rmarkdown, testthat

**Encoding** UTF-8

**Language** en-CA

**License** GPL-3

**VignetteBuilder** knitr

**ByteCompile** yes

**BugReports** <https://github.com/PredictiveEcology/fpCompare/issues>

**RoxygenNote** 7.2.1

**NeedsCompilation** no

**Author** Alex M Chubaty [aut, cre] (ORCID:

<<https://orcid.org/0000-0001-7146-8135>>),

Her Majesty the Queen in Right of Canada, as represented by the  
Minister of Natural Resources Canada [cph]

**Maintainer** Alex M Chubaty <[achubaty@for-cast.ca](mailto:achubaty@for-cast.ca)>

**Repository** CRAN

**Date/Publication** 2022-08-15 07:30:11 UTC

## Contents

<code>%&gt;=%</code> .....	2
<b>Index</b>	<b>4</b>

---

`%>=%`

*Relational operators with tolerance*

---

### Description

Binary operators which allow the comparison of values in numeric vectors.

### Usage

`x %>=% y`

`x %>>% y`

`x %<=% y`

`x %<<% y`

`x %==% y`

`x %!=% y`

### Arguments

`x` Any numeric object.

`y` Any numeric object.

### Details

These are similar to their counterparts in base, except a tolerance `fpCompare.tolerance` can be specified via options to account for floating point rounding errors:

<code>fpCompare</code>	base
<code>%&gt;=%</code>	<code>&gt;=</code>
<code>%&gt;&gt;%</code>	<code>&gt;</code>
<code>%&lt;=%</code>	<code>&lt;=</code>
<code>%&lt;&lt;%</code>	<code>&lt;</code>
<code>%==%</code>	<code>==</code>
<code>%!=%</code>	<code>!=</code>

Inspired by R FAQ 7.31 ([https://cran.r-project.org/doc/FAQ/R-FAQ.html#Why-doesn\\_0027t-R-think-these-num\\_003f](https://cran.r-project.org/doc/FAQ/R-FAQ.html#Why-doesn_0027t-R-think-these-num_003f)) and this post (<https://stackoverflow.com/a/2769618/1380598>).

**Value**

A logical vector indicating the result of the element by element comparison. The elements of shorter vectors are recycled as necessary.

**Author(s)**

Alex Chubaty

**See Also**

[all.equal](#), [.Machine](#)

**Examples**

```
x1 <- 0.5 - 0.3
x2 <- 0.3 - 0.1
x1 == x2                # FALSE on most machines
x1 %==% x2              # TRUE everywhere
identical(all.equal(x1, x2), TRUE) # TRUE everywhere
```

```
set.seed(123)
a <- 1:6
b <- jitter(1:6, 1e-7)
print(rbind(a, b), digits = 16)
```

```
b %<=% a
b %<<% a
b %>=% a
b %>>% a
b %==% a
b %!=% a
```

# Index

.Machine, 3  
%!=% (%>=%), 2  
%<% (%>=%), 2  
%<=% (%>=%), 2  
%==% (%>=%), 2  
%>% (%>=%), 2  
%>=%, 2  
all.equal, 3