fpCompare

Alex M. Chubaty

September 06, 2019

Reliable comparison of floating point numbers

Comparisons of floating point numbers are problematic due to errors associated with the binary representation of decimal numbers. Computer scientists and programmers are aware of these problems (*.e.g.*, Goldberg 1991) and yet people still use numerical methods which fail to account for floating point errors (this pitfall is the first to be highlighted in the First Circle of "The R Inferno" (Burns 2012)).

To avoid these and other numerical rounding issues, R's help file for relational operators (*e.g.*, `?'>'`) suggests using `identical` and `all.equal` when making numerical comparisons:

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

Inspired by R FAQ 7.31 and this Stack Overflow answer, this package provides new relational operators useful for performing floating point number comparisons with a set tolerance:

`fpCompare`[1]	`base`
`%>=%`	`>=`
`%>>%`	`>`
`%<=%`	`<=`
`%<<%`	`<`
`%==%`	`==`
`%!=%`	`!=`

These functions use the `base` relational operators to make comparisons, but incorporate a tolerance value (`fpCompare.tolerance`) similar to `all.equal`. The default `fpCompare.tolerance` value is `.Machine$double.eps^0.5`, set via `options`. This is the same default used in `all.equal` for numeric comparisons.

```
# set telorance value
tol = .Machine$double.eps^0.5       # default value
options(fpCompare.tolerance = tol)

# perform comparisons
x1 <- 0.5 - 0.3
x2 <- 0.3 - 0.1
x1 == x2         # FALSE on most machines
x1 %==% x2       # TRUE everywhere
```

Installation

From CRAN

```
install.packages("fpCompare")
```

From GitHub

```
library(devtools)
install_github("PredictiveEcology/fpCompare")
```

Bug Reports

https://github.com/PredictiveEcology/fpCompare/issues

References

Burns, Patrick. 2012. *The R Inferno*. 2nd ed. http://www.burns-stat.com/pages/Tutor/R_inferno.pdf.

Goldberg, David. 1991. "What every computer scientist should know about floating-point arithmetic." *ACM Computing Surveys*, 171–264. http://dl.acm.org/citation.cfm?id=103163.