Subject: Re: rounding errors
Posted by John-David T. Smith on Fri, 27 Apr 2001 15:39:21 GMT
View Forum Message <> Reply to Message

"Dominic R. Scales" wrote:
> I fully appreciate and understand the precision
> difference in significant digits between float and double, be it in idl,
> c, or any other programming language, as it is inherently diffucult
> (read impossible) to map an infinite set of numbers to a finite realm of
> 4/8 byte and not miss any.
> The case I am trying to make is this: why aren't the missing digits, i.e.
> the ones added by the cast, set to 0.? As I go along with my calculations,
> these random additional digits give me a hell of a headache by accumulating.
> And, as Murphy leads us to expect, always in the 'wrong' direction.
> I know, one shouln't test floating point numbers in digital rep. against
> one another, not even if the were double.

I think this is a case of not fully understanding how floating point numbers are stored and manipulated (it took me several tries to grasp it). When you have all those trailing zeroes, it's simply a case of the formatted printing routines trying to be clever for you. This cleverness in printout does not extend *in any way* to cleverness in calculation. It's actually not terribly intuitive.

Here is a good discussion (though for fortran): http://www.lahey.com/float.htm. Thanks to whomever originally posted that. A very relevant section:

>>>> >>> >>>>>>>
Too Many Digits

You may decide to check the previous program example by printing out both D and X in the same format, like this:

```
30  FORMAT (X, 2G25.15)
PRINT 30, X, D
```

In some FORTRAN implementations, both numbers print out the same. You may walk away satisfied, but you are actually being misled by low-order digits in the display of the single-precision number. In Lahey FORTRAN the numbers are printed out
as:

```
    1.66661000000000        1.66661000251770
```

The reason for this is fairly simple: the formatted-I/O conversion routines know that the absolute maximum decimal digits that have any significance when printing a single-precision entity is 9. The rest of

the field is filled with the current
"precision-fill" character, which is "0" by default. The precision-fill character can be changed to any ASCII character, e.g., asterisk or blank. Changing the precision-fill character to "*" emphasizes the insignificance of the low-order digits:

    1.66661000******   1.66661000251770

>>>> >> >>>>>>>

JD