Subject: Re: converting floats to doubles
Posted by Dick Jackson on Fri, 20 May 2005 20:17:57 GMT
View Forum Message <> Reply to Message

Hi,

"Benjamin Hornberger" <benjamin.hornberger@stonybrook.edu> wrote in message
news:428e3721_4@marge.ic.sunysb.edu...
> Hi computation gurus,
>
> is dblarr(n) equivalent in precision to double(fltarr(n))? I know that in
> a case like sqrt(dblarr(n)) vs. double(sqrt(fltarr(n))), they are not
> equivalent (the second version is not true double precision). But I
> thought when I start with whole numbers anyway, it might be the case.
>
> In other words, when a floating point number is converted to double, are
> the additional digits always set to zero, or is it possible that they
> aren't? I tried it out by printing some numbers, and it looks like they
> add only zeroes, but I would be happy if the experts could confirm.

You're right, it's good to be careful about these things, but indeed there
are a lot of integers that are precisely correct in Float (and even more in
Double). Empirically:

;; Run a loop until the Double version of an integer is not equal to
;; the Float version (this took several seconds to run)

IDL> for i=0D,1D9 do if i ne Double(Float(i)) then break

;; Variable 'i' (Double) has the first mismatch...

IDL> print,i,format='(F20.10)'
 16777217.0000000000
IDL> print,Float(i),format='(F20.10)'
 16777216.0000000000

Well, look at that, the number of good integer Floats is 256^3:
IDL> print,256L*256*256
    16777216
... which makes all kinds of sense, as a Float has 3 bytes for the mantissa
(or significand). For more, see:

http://en.wikipedia.org/wiki/Floating_point

Cheers,
--
-Dick

Dick Jackson / dick@d-jackson.com
D-Jackson Software Consulting / http://www.d-jackson.com
Calgary, Alberta, Canada / +1-403-242-7398 / Fax: 241-7392

Subject: Re: converting floats to doubles
Posted by Michael Wallace on Fri, 20 May 2005 20:52:03 GMT
View Forum Message <> Reply to Message

First, let me assure you that when you convert any number from a float to a double, there is absolutely no change in value. When you print out a value and do not supply a specific format, IDL will show a different number of decimal places depending on whether the number is a float or double. Here's an example of what I mean:

```
IDL> print, float(3), double(3)
      3.00000       3.0000000
```

The above results are just because IDL's default format for floats are different than the default format for doubles.

Floating point numbers do not represent a number exactly. Floating point numbers are composed of a sign bit, exponent and a mantissa. These three values are then fed into an equation which then produces the actual floating point number we see. This why I can say that converting a number from a float to a double doesn't change anything. The mantissa, exponent and sign bit of the float are copied directly into sign bit, mantissa and exponent of the double. The extra bits in the double are just left at 0.

dblarr(n) is the same as double(fltarr(n)). The fltarr(n) will create n many floating point numbers, all of which are 0. Converting all these floats into doubles will yield a double array where all values are 0 and this is the very same thing as dblarr(n).

sqrt(dblarr(n)) is not the same thing as double(sqrt(fltarr(n)). In the latter, you are taking the sqrt of the floating point numbers first. There will be precision lost because the float mantissa is only capable of storing so much information. If you take this value and cast it into a double, the less precise float value is preserved and the other bits of the double's mantissa are left at 0. Had you done sqrt(dblarr(n)), the sqrt operation would have been calculated using double precision arithmetic and the entire mantissa of the double would be filled. Because the double's mantissa is larger than the float's mantissa, it is able to store more precision.

A lot of the gory details of IEEE 754, the specification of floating point numbers, can be found here: http://en.wikipedia.org/wiki/IEEE_754.

-Mike

Benjamin Hornberger wrote:
> Hi computation gurus,
>
> is dblarr(n) equivalent in precision to double(fltarr(n))? I know that
> in a case like sqrt(dblarr(n)) vs. double(sqrt(fltarr(n))), they are not
> equivalent (the second version is not true double precision). But I
> thought when I start with whole numbers anyway, it might be the case.
>
> In other words, when a floating point number is converted to double, are
> the additional digits always set to zero, or is it possible that they
> aren't? I tried it out by printing some numbers, and it looks like they
> add only zeroes, but I would be happy if the experts could confirm.
>
> Thanks for any insight,
>
> Benjamin

Subject: Re: converting floats to doubles
Posted by Benjamin Hornberger on Fri, 20 May 2005 20:58:21 GMT
View Forum Message <> Reply to Message

Michael Wallace wrote:
>
> dblarr(n) is the same as double(fltarr(n)). The fltarr(n) will create n
> many floating point numbers, all of which are 0. Converting all these
> floats into doubles will yield a double array where all values are 0 and
> this is the very same thing as dblarr(n).

Actually I meant dindgen(n) vs. double(findgen(n)) in the first place. I always mix them up ... But I think I got the point.

Thanks,
Benjamin

Subject: Re: converting floats to doubles
Posted by Michael Wallace on Fri, 20 May 2005 21:35:20 GMT
View Forum Message <> Reply to Message

>> dblarr(n) is the same as double(fltarr(n)). The fltarr(n) will create
>> n many floating point numbers, all of which are 0. Converting all
>> these floats into doubles will yield a double array where all values

>> are 0 and this is the very same thing as dblarr(n).
>
>
> Actually I meant dindgen(n) vs. double(findgen(n)) in the first place. I
> always mix them up ... But I think I got the point.

Ah, okay. Same logic still applies.

Because doubles have the same structure as floats, but more capacity, the set of all possible double precision values on a particular architecture is a superset of the set of all possible single precision (float) values on the same architecture. That, in short, was what all my rambling before was trying to say. I have never been one to really understand what the word "concise" means. :-)

-Mike

Subject: Re: converting floats to doubles
Posted by Peter Mason on Sun, 22 May 2005 23:01:23 GMT
View Forum Message <> Reply to Message

Benjamin Hornberger wrote:
<...>
> Actually I meant dindgen(n) vs. double(findgen(n)) in the first
> place. I always mix them up ... But I think I got the point.

Hi Benjamin,

A single-precision float (as in findgen()) has a mantissa that's effectively 24 bits. Consequentially it represents every integer in the range [-16777215, 16777215] *exactly*. So if your "n" in "findgen(n)" is smaller than 1677216, it doesn't matter whether you use double(findgen(n)) or dindgen(n).
For bigger "n", you will start getting repeat values if you use double(findgen(n)) as the float mantissa will have run out of bits and the double conversion happens after the fact. Here, dindgen() is better. The double mantissa is much bigger (effectively 53 bits) so dindgen ~should~ only start hitting repeats for "n" >= 2^53. (I can't actually check if the algorithm within dindgen() behaves properly outside the 32-bit unsigned integer range. Dindgen(2LL^32LL) alone would be a 32GB array! :-))

Peter Mason
