Subject: Re: Math Question
Posted by Foldy Lajos on Sun, 29 Oct 2006 17:59:50 GMT
View Forum Message <> Reply to Message

On Sun, 29 Oct 2006, David Fanning wrote:

> Folks,
>
> OK, I get the feeling that I am going to be referred to
> my own web page with this question:
>
> http://www.dfanning.com/math_tips/sky_is_falling.html
>
> And it is certainly true that I have been watching WAY
> too much TV lately (World Series, you know), but here is
> my question. How does one explain the following two
> IDL commands?
>
> IDL> Help, (-0.1)^2.0
> <Expression> FLOAT = 0.0100000
> IDL> Help, (-0.1)^2.01
> <Expression> FLOAT = -NaN
>
> In general, raising a negative number to any integer
> power seems to produce a real number, whereas raising
> a negative number to a non-integer power causes a NAN.
>
> I am sure this is explained in one of those textbooks
> covered with dust in my garage, but I thought one of
> you math guys could rescue me from a beautiful day
> spent covered with dust. :-)
>
> Cheers,
>
> David
> --
> David Fanning, Ph.D.
> Fanning Software Consulting, Inc.
> Coyote's Guide to IDL Programming: http://www.dfanning.com/
> Sepore ma de ni thui. ("Perhaps thou speakest truth.")
>

I had the opposite problem with FL: (-0.1)^2.0 was also a NaN. Power is defined for floats as x^y = exp(log(x)*y), and log() is undefined for non-positive numbers. Now FL examines y first, and if its is an integer, then uses multiplications. I think IDL does the same.

regards,

lajos

ps: what will be the result for (-0.1)^2.0000001 ? :-)

Subject: Re: Math Question
Posted by Robbie on Sun, 29 Oct 2006 23:01:44 GMT
View Forum Message <> Reply to Message

Consider

y = x^(a/b)

then

y^b = x^a

The result of a and y being real happens when a+b is even. I have some scrawl on the back of an envelope, but the reasoning is probably buried in that dusty textbook somewhere.

If we are using floating point arithmetic, then we can't really find out what a or b is because the accuracy of the machine is not perfect.

i.e. 2.01 = 201/200 - but we really can't be sure.

Subject: Re: Math Question
Posted by Robbie on Sun, 29 Oct 2006 23:40:06 GMT
View Forum Message <> Reply to Message

Actually that reasoning is totally flawed, we should keep those envelopes for experimental physicists.

Lookup "Mathematical Operators", subheading "Using Exponentiation" in the IDL help.

Subject: Re: Math Question
Posted by Rob Dimeo on Mon, 30 Oct 2006 00:15:15 GMT
View Forum Message <> Reply to Message

In general, powers of negative numbers are complex so you should start with that assumption in the expression:

```
IDL> print,(dcomplex(-1.0,0.0))^2.01
(     0.99950656,     0.031410729)
```

IDL> print,(dcomplex(-1.0,0.0))^2.0
(1.0000000, -2.4492127e-016)

It would be nice if IDL told you this rather than throwing a NaN at
you.

Rob

On Oct 29, 11:02 am, David Fanning <n...@dfanning.com> wrote:
> Folks,
>
> OK, I get the feeling that I am going to be referred to
> my own web page with this question:
>
>   http://www.dfanning.com/math_tips/sky_is_falling.html
>
> And it is certainly true that I have been watching WAY
> too much TV lately (World Series, you know), but here is
> my question. How does one explain the following two
> IDL commands?
>
>   IDL> Help, (-0.1)^2.0
>   <Expression>   FLOAT   =   0.0100000
>   IDL> Help, (-0.1)^2.01
>   <Expression>   FLOAT   =   -NaN
>
> In general, raising a negative number to any integer
> power seems to produce a real number, whereas raising
> a negative number to a non-integer power causes a NAN.
>
> I am sure this is explained in one of those textbooks
> covered with dust in my garage, but I thought one of
> you math guys could rescue me from a beautiful day
> spent covered with dust. :-)
>
> Cheers,
>
> David
> --
> David Fanning, Ph.D.
> Fanning Software Consulting, Inc.
> Coyote's Guide to IDL Programming:http://www.dfanning.com/
> Sepore ma de ni thui. ("Perhaps thou speakest truth.")

Subject: Re: Math Question
Posted by edward.s.meinel@aero. on Mon, 30 Oct 2006 15:19:01 GMT

Rob wrote:
> In general, powers of negative numbers are complex so you should start
> with that assumption in the expression:
>
> IDL> print,(dcomplex(-1.0,0.0))^2.01
> (0.99950656, 0.031410729)
> IDL> print,(dcomplex(-1.0,0.0))^2.0
> (1.0000000, -2.4492127e-016)
>
> It would be nice if IDL told you this rather than throwing a NaN at
> you.

Actually, to be consistent, IDL should just return a complex number.
One of the reasons that I've been using IDL is that it automatically
changes the variable TYPE when necessary so that I don't have to keep
track of whether I'm operating on a BYTE, LONG, DOUBLE, whatever... Now
I find out that it's "mostly" true (but who can refuse a nice mutton
sandwich?).

I would consider this a bug.

Ed "to blave" Meinel

Subject: Re: Math Question
Posted by Foldy Lajos on Mon, 30 Oct 2006 15:29:36 GMT

On Mon, 30 Oct 2006, edward.s.meinel@aero.org wrote:

>
> Rob wrote:
>> In general, powers of negative numbers are complex so you should start
>> with that assumption in the expression:
>>
>> IDL> print,(dcomplex(-1.0,0.0))^2.01
>> (0.99950656, 0.031410729)
>> IDL> print,(dcomplex(-1.0,0.0))^2.0
>> (1.0000000, -2.4492127e-016)
>>
>> It would be nice if IDL told you this rather than throwing a NaN at
>> you.
>
> Actually, to be consistent, IDL should just return a complex number.
> One of the reasons that I've been using IDL is that it automatically
> changes the variable TYPE when necessary so that I don't have to keep

> track of whether I'm operating on a BYTE, LONG, DOUBLE, whatever... Now
> I find out that it's "mostly" true (but who can refuse a nice mutton
> sandwich?).
>
> I would consider this a bug.
>
> Ed "to blave" Meinel
>

I think complex results should be returned only if complex input was
given, like in the example above. Otherwise, all your data would easily
become complex. Think of sqrt(-1.0), log(-1.0), asin(-2.0), etc.

regards,
lajos

Subject: Re: Math Question
Posted by news.qwest.net on Mon, 30 Oct 2006 16:54:13 GMT
View Forum Message <> Reply to Message

"David Fanning" <news@dfanning.com> wrote in message
news:MPG.1fae7e91b1ccf3e3989d78@news.frii.com...
> Folks,
...
> In general, raising a negative number to any integer
> power seems to produce a real number, whereas raising
> a negative number to a non-integer power causes a NAN.

Evidently IDL needs another value, NaRN (not a real number).

Cheers,
bob

Subject: Re: Math Question
Posted by James Kuyper on Mon, 30 Oct 2006 17:04:46 GMT
View Forum Message <> Reply to Message

Rob wrote:
> In general, powers of negative numbers are complex so you should start
> with that assumption in the expression:
>
> IDL> print,(dcomplex(-1.0,0.0))^2.01
> (0.99950656, 0.031410729)
> IDL> print,(dcomplex(-1.0,0.0))^2.0

> (1.0000000, -2.4492127e-016)
>
> It would be nice if IDL told you this rather than throwing a NaN at
> you.
>
> Rob
>
> On Oct 29, 11:02 am, David Fanning <n...@dfanning.com> wrote:
>> Folks,
>>
>> OK, I get the feeling that I am going to be referred to
>> my own web page with this question:
>>
>> http://www.dfanning.com/math_tips/sky_is_falling.html
>>
>> And it is certainly true that I have been watching WAY
>> too much TV lately (World Series, you know), but here is
>> my question. How does one explain the following two
>> IDL commands?
>>
>> IDL> Help, (-0.1)^2.0
>> <Expression> FLOAT = 0.0100000
>> IDL> Help, (-0.1)^2.01
>> <Expression> FLOAT = -NaN
>>
>> In general, raising a negative number to any integer
>> power seems to produce a real number, whereas raising
>> a negative number to a non-integer power causes a NAN.
>>
>> I am sure this is explained in one of those textbooks
>> covered with dust in my garage, but I thought one of
>> you math guys could rescue me from a beautiful day
>> spent covered with dust. :-)

If a and b are mutually prime integers, then mathematically, x^(a/b) has b different values, at most two of which are real, the others are complex. If b is odd, only one of the values is real. This is true, whether or not 'x' is negative. However, I don't think we want to have (-0.10)^2.01 return a list of 200 different complex values. Even when the operands are complex number IDL only attempts to return one of the possible values.

More imporatantly, I don't think it should return any complex values at all. In most cases, if the operands of the ^ operator aren't already complex, there's a pretty good chance that the code wasn't written to handle the possiblity of complex values from the result.

Considered solely as a function whose domain and range are restricted

to real numbers, x^(a/b), should have two solutions if b is even, and 1 solution if b is odd, regardless of whether x is positive. In practice, for computer arithmetic that's not generally implementable. When you calculate x^y, y can, in general, only be a floating point approximation of a/b; it is an exact representation only if b is a power of 2. I've used implementations of the power function that attempt to recognize when y is an approximation of a/b for small values of b, however, I gather that IDL is implemented using C code, expwhich probably calls pow(x,y) to implement x^y. The pow() function does not attempt to recognise approximations to rational numbers with small denominators. It is required by the C standard to return a domain error if x is negative and y is a finite non-integer. IDL handles this by returning a NaN. This strikes me as a reasonable approach.

Subject: Re: Math Question
Posted by David Fanning on Mon, 30 Oct 2006 17:19:41 GMT
View Forum Message <> Reply to Message

Folks (and especially Mr. Kuyper):

Much obliged for the discussion. Most enlightening. :-)

Cheers,

David

P.S. You will be pleased to know I put the pretty
day to good advantage and won all four sets of tennis!
Lots of sweat, but no dust to be found. :-)

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Sepore ma de ni thui. ("Perhaps thou speakest truth.")

Subject: Re: Math Question
Posted by James Kuyper on Mon, 30 Oct 2006 18:15:47 GMT
View Forum Message <> Reply to Message

I shouldn't attempt to remember advanced mathematics in public when I'm less than fully awake. :-(It's been a long time since I actually used any of this stuff.

kuyper@wizard.net wrote:

...
> Considered solely as a function whose domain and range are restricted
> to real numbers, x^(a/b), should have two solutions if b is even, and 1
> solution if b is odd, regardless of whether x is positive.

That should be "two solutions if b is even and x is positive, and 1
solution if b is odd, regardless of whether x is positive".

After realizing that mistake, I decided to double check my statement
about the case where the range includes complex values:

...
> If a and b are mutually prime integers, then mathematically, x^(a/b)
> has b different values, at most two of which are real, the others are
> complex. If b is odd, only one of the values is real. This is true,

That count of real values is based upon the assumption that the
imaginary part of x is 0, and that x is positive. There's no simple
statement that covers the general case.

Subject: Re: Math Question
Posted by Braedley on Mon, 30 Oct 2006 18:22:54 GMT
View Forum Message <> Reply to Message

edward.s.meinel@aero.org wrote:
> Rob wrote:
>> In general, powers of negative numbers are complex so you should start
>> with that assumption in the expression:
>>
>> IDL> print,(dcomplex(-1.0,0.0))^2.01
>> (0.99950656, 0.031410729)
>> IDL> print,(dcomplex(-1.0,0.0))^2.0
>> (1.0000000, -2.4492127e-016)
>>
>> It would be nice if IDL told you this rather than throwing a NaN at
>> you.
>
> Actually, to be consistent, IDL should just return a complex number.
> One of the reasons that I've been using IDL is that it automatically
> changes the variable TYPE when necessary so that I don't have to keep
> track of whether I'm operating on a BYTE, LONG, DOUBLE, whatever... Now
> I find out that it's "mostly" true (but who can refuse a nice mutton
> sandwich?).
>
> I would consider this a bug.
>
> Ed "to blave" Meinel

Just out of curiosity, has anyone tried this on Matlab? I'd expect the same (or similar) results, but it'd be interesting if they weren't.

Braedley

Subject: Re: Math Question
Posted by Jo Klein on Mon, 30 Oct 2006 20:17:55 GMT
View Forum Message <> Reply to Message

> Just out of curiosity, has anyone tried this on Matlab? I'd expect the
> same (or similar) results, but it'd be interesting if they weren't.
Matlab returns the real part of the argument if it's a real number you put in:
>> -0.1^2.0

ans =
-0.0100

>> -0.1^2.01
ans =
-0.0098

>> complex(-0.1,0)^2.01
ans =
0.0098 + 0.0003i
Hmm - I don't know if this is more desirable than IDL's behaviour, I think it's fair enough to warn people who try to do something like that with their data. In Matlab, try and invert the second operation ... this can't be good. Suppose there are arguments for both approaches.
Jo

Subject: Re: Math Question
Posted by Jean H. on Mon, 30 Oct 2006 20:32:15 GMT
View Forum Message <> Reply to Message

Jo Klein wrote:
>> Just out of curiosity, has anyone tried this on Matlab? I'd expect the
>> same (or similar) results, but it'd be interesting if they weren't.
>
> Matlab returns the real part of the argument if it's a real number you
> put in:
>>> -0.1^2.0
>
> ans =

>   -0.0100
>
>>> -0.1^2.01
> ans =
>   -0.0098
>
>>> complex(-0.1,0)^2.01
> ans =
>   0.0098 + 0.0003i
> Hmm - I don't know if this is more desirable than IDL's behaviour, I
> think it's fair enough to warn people who try to do something like that
> with their data. In Matlab, try and invert the second operation ... this
> can't be good. Suppose there are arguments for both approaches.
> Jo

It depends indeed...

 >> -0.1^2.1
ans =
   -0.0079

 >> (-0.1)^2.1
ans =
   0.0076 + 0.0025i

You don't have to specify the complex() statement!

Jean

Subject: Re: Math Question
Posted by Jean H. on Mon, 30 Oct 2006 20:35:57 GMT
View Forum Message <> Reply to Message

Jean H. wrote:
> Jo Klein wrote:
>
>>> Just out of curiosity, has anyone tried this on Matlab? I'd expect the
>>> same (or similar) results, but it'd be interesting if they weren't.
>>
>>
>> Matlab returns the real part of the argument if it's a real number you
>> put in:
>>>> -0.1^2.0
>>
>> ans =
>>   -0.0100
>>

>>>> -0.1^2.01
>> ans =
>>   -0.0098
>>
>>>> complex(-0.1,0)^2.01
>> ans =
>>   0.0098 + 0.0003i
>> Hmm - I don't know if this is more desirable than IDL's behaviour, I
>> think it's fair enough to warn people who try to do something like
>> that with their data. In Matlab, try and invert the second operation
>> ... this can't be good. Suppose there are arguments for both approaches.
>> Jo
>
>
> It depends indeed...
>
>>> -0.1^2.1
> ans =
>   -0.0079
>
>>> (-0.1)^2.1
> ans =
>   0.0076 + 0.0025i
>
> You don't have to specify the complex() statement!
>
> Jean

for clarity:
 >> a=-0.1
a =
   -0.1000

 >> a^2
ans =
    0.0100

 >> a^2.1
ans =
   0.0076 + 0.0025i

Subject: Re: Math Question
Posted by news.qwest.net on Mon, 30 Oct 2006 21:07:57 GMT
View Forum Message <> Reply to Message

"Jo Klein" <jo_kln@yahoo.co.uk> wrote in message
news:ei5mlj$sd3$1@frank-exchange-of-views.oucs.ox.ac.uk...

>> Just out of curiosity, has anyone tried this on Matlab? I'd expect the
>> same (or similar) results, but it'd be interesting if they weren't.
> Matlab returns the real part of the argument if it's a real number you put
> in:
>>> -0.1^2.0
>
> ans =
> -0.0100

I think you have an order of operations problem there.
Matlab has given you
>> -(0.1^2.0)

instead of the intended
>> (-0.1)^2.0

likewise in the other case

Cheers,
bob

Subject: Re: Math Question
Posted by Jo Klein on Tue, 31 Oct 2006 12:33:24 GMT
View Forum Message <> Reply to Message

>> It depends indeed...
>>
>>>> -0.1^2.1
>> ans =
>> -0.0079
>>
>>>> (-0.1)^2.1
>> ans =
>> 0.0076 + 0.0025i

Ah, my mistake - operator precedence strikes again. The first statement
actually expands to something like 0-(0.1^2.1). D'uh.
Jo

Subject: Re: Math Question
Posted by rkombiyil on Wed, 01 Nov 2006 12:51:01 GMT
View Forum Message <> Reply to Message

And fwiw, here is what OCTAVE gives :-)

--
>> -0.1^2
ans = -0.010000
>> -0.1^2.1
ans = -0.0079433
>> .1^2.1
ans = 0.0079433
>> (-0.1)^2.1
ans = 0.0075545 + 0.0024546i
--
Cheers,
~rk
