Subject: Re: unwrap modulo 2pi
Posted by graham_wilson on Wed, 07 Feb 2001 15:30:49 GMT
View Forum Message <> Reply to Message

(snip)
>>
>> x = 2*seq(5)
>> y = 2*Pi
>> x=mod(x,y)
>> x=colunmod(x,y)
>> print x
>
> What are 'seq' and 'colunmod'?
> It seems you have 1D data. Is it right?
> In this case you'll find several routine searching on the IDL library
> on the net.

The 'seq' command simply makes a vector [2,4,6,8,10] and the 'colunmod' function is what I am looking for i.e. a function that will undo/unwrap the function mod(x,y). I have looked for this function on the net as you suggested (Martin's IDL 11 online libraries) but I have yet to find it... Can you be more specific regarding which library or where I should look?

Indeed, I only need the dim 1 case.

Thanks!
GW

Sent via Deja.com
http://www.deja.com/

Subject: Re: unwrap modulo 2pi
Posted by Pavel A. Romashkin on Wed, 07 Feb 2001 17:02:09 GMT
View Forum Message <> Reply to Message

I am sorry, but I am still a little behind here, please bear with me. Must be the lack of coffe. What is the "unwrapping of the function mod(x,y)" ? I might think of a solution if I knew what I am looking at.

Cheers,
Pavel

graham_wilson@my-deja.com wrote:

> The 'seq' command simply makes a vector [2,4,6,8,10] and the 'colunmod'

> function is what I am looking for i.e. a function that will undo/unwrap
> the function mod(x,y). I have looked for this function on the net as
> you suggested (Martin's IDL 11 online libraries) but I have yet to find
> it... Can you be more specific regarding which library or where I should
> look?
>
> Indeed, I only need the dim 1 case.
>
> Thanks!
> GW
>
> Sent via Deja.com
> http://www.deja.com/

Subject: Re: unwrap modulo 2pi
Posted by graham_wilson on Wed, 07 Feb 2001 17:52:14 GMT
View Forum Message <> Reply to Message

My appologies for not being explicit enough...

```
IDL> a=[2,4,6,8,10,12]
IDL> a=[2.,4.,6.,8.,10.,12.]
IDL> b=2*!PI
IDL> c=a mod b
IDL> print, c
      2.00000  4.00000   6.00000   1.71681  3.71681   5.71681
```

What I mean by 'unwraping' is: Given I know 'c' and 'b' how do I
explicitly find a?

GW

In article <3A817F92.1DBCDCCA@noaa.gov>,
"Pavel A. Romashkin" <pavel.romashkin@noaa.gov> wrote:
> I am sorry, but I am still a little behind here, please bear with me.
> Must be the lack of coffe. What is the "unwrapping of the function
> mod(x,y)" ? I might think of a solution if I knew what I am looking
at.
>
> Cheers,
> Pavel

Sent via Deja.com
http://www.deja.com/

Subject: Re: unwrap modulo 2pi
Posted by Dave Greenwood on Wed, 07 Feb 2001 18:29:12 GMT
View Forum Message <> Reply to Message

graham_wilson@my-deja.com wrote:
> My appologies for not being explicit enough...
>
> IDL> a=[2,4,6,8,10,12]
> IDL> a=[2.,4.,6.,8.,10.,12.]
> IDL> b=2*!PI
> IDL> c=a mod b
> IDL> print, c
> 2.00000 4.00000 6.00000 1.71681 3.71681 5.71681
>
> What I mean by 'unwraping' is: Given I know 'c' and 'b' how do I
> explicitly find a?

Surely I must be missing something(?):

```
 IDL> print, c + fix(a/b)*b
     2.00000      4.00000      6.00000      8.00000      10.0000      12.0000
```

Dave

Dave Greenwood Email: Greenwoodde@ORNL.GOV
Oak Ridge National Lab %STD-W-DISCLAIMER, I only speak for myself

Subject: Re: unwrap modulo 2pi
Posted by Dave Greenwood on Wed, 07 Feb 2001 18:48:25 GMT
View Forum Message <> Reply to Message

I wrote:
> graham_wilson@my-deja.com wrote:
>> My appologies for not being explicit enough...
>>
>> IDL> a=[2,4,6,8,10,12]
>> IDL> a=[2.,4.,6.,8.,10.,12.]
>> IDL> b=2*!PI
>> IDL> c=a mod b
>> IDL> print, c
>> 2.00000 4.00000 6.00000 1.71681 3.71681 5.71681
>>
>> What I mean by 'unwraping' is: Given I know 'c' and 'b' how do I
>> explicitly find a?
>
> Surely I must be missing something(?):
>

> IDL> print, c + fix(a/b)*b
> 2.00000 4.00000 6.00000 8.00000 10.0000 12.0000

And a proverbial ohnosecond later I went - "Oh No!". I guess I better
go back to lurking. ;-(

Btw, given that (a mod b) = (a+(n*b) mod b) do you expect your 'unwraping'
function to give the lowest value of a such that c = a mod b?

Dave

Subject: Re: unwrap modulo 2pi
Posted by tam on Wed, 07 Feb 2001 19:18:30 GMT
View Forum Message <> Reply to Message

Perhaps I'm confused but it seems to me you have
a misunderstanding of what the modulo function does.
Since you indicate that omatrix has an 'unwrapping'
function I figure I'm still missions something...

If
  c = a mod b

and I have the values of c and b, then all I know about
a -- absent any other information -- is that

  a element_of {c + n*b}

where n ranges over the integers. [Some languages
treat modulos of negative numbers differently so
we might be restricted to n >= 0). The modulus function
by definition gives values that repeat with a cycle
length of b, so the easiest solution for an
inverse of it is simply the identity function.
I.e., taking your fourth example.

  8 mod 2*!pi = 1.71681
but
  1.71681 mod 2*!pi = 1.71681

You've lost information using the mod function and
you can get it back...

The one thing I'm guessing is that in omatrix your
'unwrapping' function works on arrays and requires
that the value of the array increase monotonically.
Presumably the delta's between the array values

are not constant -- or the unwrapping is trivial --
so one is still not guaranteed that the unwrapping
proceeds correctly. However in the case where the spacing
between array values is never greater than b you should be able
to write a relatively simple unwrapping function...

Something like [untested and I've no doubt
someone can do it without loops.]:

```
   function unwrap, a, modval

      len  = n_elements(a)
      if (len lt 2) then begin  ; Need to have multiple elements
         b = a
         return, b
      endif

      w    = where (a[0:len-2] gt a[1:len-1])  ; Find the wrapping points
      if (w[0] eq -1) then begin              ; If none just return a copy
of input
         b = a
         return, b
      endif

      z = a          ; Get a copy of the input
      z[*] = 0       ; Set all values to nil
      z[w] = modval   ; Set deltas at wrapping points.
      for i=1, n_elements(z)-1 do begin  ; Loop to add deltas.
        z[i] = z[i] + z[i-1]
      endfor

      return, a + z   ; Add the deltas back in.
   end
```

Again I note that this inversion of the modulus is possible only
given the additional information that the input array was monotonic
and had sufficiently small increments. If your additional information
is different, then a different inverse may be appropriate (e.g.,
if you know the input values are integral). In general the function
is not invertible.

Regards,
Tom McGlynn

graham_wilson@my-deja.com wrote:

>
> My appologies for not being explicit enough...
>
> IDL> a=[2,4,6,8,10,12]
> IDL> a=[2.,4.,6.,8.,10.,12.]
> IDL> b=2*!PI
> IDL> c=a mod b
> IDL> print, c
> 2.00000 4.00000 6.00000 1.71681 3.71681 5.71681
>
> What I mean by 'unwraping' is: Given I know 'c' and 'b' how do I
> explicitly find a?
>
> GW
>
> In article <3A817F92.1DBCDCCA@noaa.gov>,
> "Pavel A. Romashkin" <pavel.romashkin@noaa.gov> wrote:
>> I am sorry, but I am still a little behind here, please bear with me.
>> Must be the lack of coffe. What is the "unwrapping of the function
>> mod(x,y)" ? I might think of a solution if I knew what I am looking
> at.
>>
>> Cheers,
>> Pavel
>
> Sent via Deja.com
> http://www.deja.com/

Subject: Re: unwrap modulo 2pi
Posted by Randall Skelton on Wed, 07 Feb 2001 19:48:10 GMT
View Forum Message <> Reply to Message

Modulo Operator Notes

quotient: q = a / b
remainder: r = a mod b

So,

a	b	a/b	a mod b
10	3	3	1
3	10	0	3
-10	-3	-3	-1

It is *always* true that a = q*b + r with abs(r) < abs(b) and b neq 0

Randall

graham_wilson@my-deja.com wrote:
> My appologies for not being explicit enough...
>
> IDL> a=[2,4,6,8,10,12]
> IDL> a=[2.,4.,6.,8.,10.,12.]
> IDL> b=2*!PI
> IDL> c=a mod b
> IDL> print, c
> 2.00000 4.00000 6.00000 1.71681 3.71681 5.71681
>
> What I mean by 'unwraping' is: Given I know 'c' and 'b' how do I
> explicitly find a?
>

Subject: Re: unwrap modulo 2pi
Posted by Pavel A. Romashkin on Wed, 07 Feb 2001 20:39:11 GMT
View Forum Message <> Reply to Message

I just don't see that you can do this at all. Modulo operator discards information about the number of multiples of B from A, leaving only the remainder of the division operation. For example, if:

c = [2., 4., 6., 1.71681, 3.71681, 5.71681]
b = 2*!PI

; Each of the following A will produce the given
; C through "C=A mod B" :

a=[2.,4.,6.,8.,10.,12.]
a=[2.,4.,6.,8.,10.,12.] + b
a=[2.,4.,6.,8.,10.,12.] +2*b

a=[2.,4.,6.,8.,10.,12.] & a[3:5] = a[3:5] + 3*b

Unless I am missing something, I see no way how a unique solution can be obtained from the *remainder of division* and *divisor*. But I studied arithmetics a long time ago :-)

Cheers,
Pavel

graham_wilson@my-deja.com wrote:
>
> My appologies for not being explicit enough...
>
> IDL> a=[2,4,6,8,10,12]

> IDL> a=[2.,4.,6.,8.,10.,12.]
> IDL> b=2*!PI
> IDL> c=a mod b
> IDL> print, c
>   2.00000 4.00000   6.00000   1.71681 3.71681   5.71681
>
> What I mean by 'unwraping' is: Given I know 'c' and 'b' how do I
> explicitly find a?
>
> GW
>
> In article <3A817F92.1DBCDCCA@noaa.gov>,
>   "Pavel A. Romashkin" <pavel.romashkin@noaa.gov> wrote:
>> I am sorry, but I am still a little behind here, please bear with me.
>> Must be the lack of coffe. What is the "unwrapping of the function
>> mod(x,y)" ? I might think of a solution if I knew what I am looking
> at.
>>
>> Cheers,
>> Pavel
>
> Sent via Deja.com
> http://www.deja.com/

Subject: Re: unwrap modulo 2pi
Posted by Craig Markwardt on Wed, 07 Feb 2001 21:07:03 GMT
View Forum Message <> Reply to Message

"Pavel A. Romashkin" <pavel.romashkin@noaa.gov> writes:

> I just don't see that you can do this at all. Modulo operator discards
> information about the number of multiples of B from A, leaving only the
> remainder of the division operation. For example, if:
>
... deleted ...
>
> Unless I am missing something, I see no way how a unique solution can be
> obtained from the *remainder of division* and *divisor*. But I studied
> arithmetics a long time ago :-)

Pavel, and others--

He is looking for an algorithm that can *reconstruct* the full number.
Of course this will use local information from the surrounding points.

His example is perhaps a little too simple. Try this one:

ph = findgen(100)
ph1 = atan(sin(ph), cos(ph))

Now, PH and PH1 represent the same phase angle on the circle, but PH1
has the disadvantage of being a discontinuous sawtooth function.
Sometimes you want to reconstruct PH based only on knowledge of PH1.
Assuming that the function is nearly monotonic and there is never a
phase jump more than !dpi, I believe this can be done.

The naive solution (which I've never gotten beyond) is to look for
discontinuities in PH1 of (say) more than !dpi. When such a
discontinuity is found, assume we have wound around once, so add
another 2*!dpi to the number.

Craig

--
---------- ----------
Craig B. Markwardt, Ph.D. EMAIL: craigmnet@cow.physics.wisc.edu
Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response
---------- ----------

Subject: Re: unwrap modulo 2pi
Posted by Pavel A. Romashkin on Wed, 07 Feb 2001 23:25:37 GMT
View Forum Message <> Reply to Message

Craig Markwardt wrote:
>
> Now, PH and PH1 represent the same phase angle on the circle, but PH1
> has the disadvantage of being a discontinuous sawtooth function.
> Sometimes you want to reconstruct PH based only on knowledge of PH1.
> Assuming that the function is nearly monotonic and there is never a
> phase jump more than !dpi, I believe this can be done.

Far beyond my primitive computational abilities :-(

Pavel

Subject: Re: unwrap modulo 2pi
Posted by Wim Bouwman on Thu, 08 Feb 2001 09:13:28 GMT
View Forum Message <> Reply to Message

In the following bit of code the measured phase P is unwrapped onto
CALCPhi.
This is done by doing a 2nd order extrapolation of the three previous

points. It works when the jumps are not too extreme.

```
;Correct Phi and PhiOne for 2Pi jumps. The calculated Phi is compared to
;the experimentally obtained Phi. Extrapolation with least squares fit.
NPhi=N_ELEMENTS(P)
CALCphi=DBLARR(NPhi)
FOR i=3,(NPhi-1) DO BEGIN
 CALCphi(i)=(-2*P(i-3)+P(i-2)+4*P(i-1))/3
 Njumps=NINT((CALCphi(i)-P(i))/(2*!DPi), /Long)
 P(i)=P(i)+Njumps*2*!DPi
ENDFOR
```

graham_wilson@my-deja.com wrote:

```
>  IDL> a=[2,4,6,8,10,12]
>  IDL> a=[2.,4.,6.,8.,10.,12.]
>  IDL> b=2*!PI
>  IDL> c=a mod b
>  IDL> print, c
>       2.00000  4.00000   6.00000   1.71681  3.71681   5.71681
>
>  What I mean by 'unwraping' is: Given I know 'c' and 'b' how do I
>  explicitly find a?
```

--

Dr. Wim G. Bouwman phone (++31) (0)15 2786775
Interfacultair Reactor Instituut fax (++31) (0)15 2788303
Technische universiteit Delft w.g.bouwman@iri.tudelft.nl
Mekelweg 15 http://www.iri.tudelft.nl/~bouwman
2629 JB Delft The Netherlands

Subject: Re: unwrap modulo 2pi
Posted by Craig Markwardt on Thu, 08 Feb 2001 15:43:20 GMT
View Forum Message <> Reply to Message

"Wim G. Bouwman" <w.g.bouwman@iri.tudelft.nl> writes:

```
>  Dear Graham,
>  The function is written in PV-WAVE and NINT is a standard function giving
>  the integer value as a long.  I am sure that something similar does exist
>  in IDL.
>  In PV-WAVE you have also FIX, which returns the integer value as an
>  integer.
```

Just curious, how does NINT do the rounding? In IDL there are FLOOR,
CEIL and ROUND which have well-behaved rounding properties (round up,

down and to-nearest, respectively). The well-known LONG function is
not consistent depending on the sign of its argument.

Craig

--
---------- ----------
Craig B. Markwardt, Ph.D. EMAIL: craigmnet@cow.physics.wisc.edu
Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response
---------- ----------

Subject: Re: unwrap modulo 2pi
Posted by Wim Bouwman on Fri, 09 Feb 2001 08:42:43 GMT
View Forum Message <> Reply to Message

> Just curious, how does NINT do the rounding? In IDL there are FLOOR,
> CEIL and ROUND which have well-behaved rounding properties (round up,
> down and to-nearest, respectively). The well-known LONG function is
> not consistent depending on the sign of its argument.

A copy of the NINT-page of the pv-wave manual:

NINT
Converts input to the nearest integer.

Usage
result = NINT(x)

Input Parameters
x : A scalar or array of any PV-WAVE variable type, usually float or double.

Keywords
Long : If present and non-zero, NINT returns a long instead of a short (FIX)
integer.

Returned Value
result : The nearest integer to the input value.

Discussion
Instead of truncating the input (as FIX does), first the input is rounded by
adding or subtracting 0.5 (depending on whether the input is greater or less
than zero), and then it is truncated.
If the input is out of the range of integers (for example, if you pass in
1.0d33), an error message will result and NINT returns garbage.
Add ï¿½0.5 to the input and convert that to a short integer using FIX. If the
Long keyword is used, it's converted via long. If the input is a FIX, then it's
just passed back. If it's a long, it's also passed back. Strings are converted

to bytes before the rounding. In the case of complex values, their magnitude is taken. Structures are not allowed.

--

Dr. Wim G. Bouwman phone (++31) (0)15 2786775
Interfacultair Reactor Instituut fax (++31) (0)15 2788303
Technische universiteit Delft w.g.bouwman@iri.tudelft.nl
Mekelweg 15 http://www.iri.tudelft.nl/~bouwman
2629 JB Delft The Netherlands