
Subject: PLOT3D format input

Posted by [lucsmm](#) on Wed, 18 Mar 2015 01:04:21 GMT

[View Forum Message](#) <> [Reply to Message](#)

Hello

I am using PLOT3D

and when the format of my x,y,z vectors is

X FLOAT = Array[200]

Y FLOAT = Array[200]

Z FLOAT = Array[200]

it works

Now, when the format of the vectors is

X FLOAT = Array[1, 108]

Y FLOAT = Array[1, 108]

Z FLOAT = Array[1, 108]

it doesn't work and IDL says that

% Expression must be a scalar or 1 element array in this context: <BYTE Array[2]>

any idea what is happening, my original data is in a (9,108) array. how do I convert them ... this is similar to my code, t is generated from other software

```
t=FINDGEN(5,200)
```

```
x = COS(t) * (1 + t / 10)
```

```
help, x
```

```
p = PLOT3D(x(0,*), x(1,*), x(2,*),'o')
```

```
t=FINDGEN(5,200)
```

```
x = t(1,*)
```

```
y = t(2,*)
```

```
z = t(3,*)
```

```
help, x,y,z
```

```
p = PLOT3D(x, y, z,'o')
```

Subject: Re: PLOT3D format input

Posted by [Matthew Argall](#) on Wed, 18 Mar 2015 02:49:46 GMT

[View Forum Message](#) <> [Reply to Message](#)

```
> t=FINDGEN(5,200)
```

```
> x = t(1,*)
```

```
> y = t(2,*)
```

```
> z = t(3,*)
```

```
>
> p = PLOT3D(x, y, z,'o')
```

I have found that function graphics are particularly picky about whether you pass them a column vector or row vector. Problems arise when you pass data in as a row vector (a 1xN array, as you are doing above). The solution is to reform your data into a column vector (Nx1 array), like so

```
x = reform(t[1,*])
y = reform(t[2,*])
z = reform(t[3,*])
```

```
p = plot3d(x, y, z, 'o')
```

Subject: Re: PLOT3D format input

Posted by [David Fanning](#) on Wed, 18 Mar 2015 03:17:38 GMT

[View Forum Message](#) <> [Reply to Message](#)

Matthew Argall writes:

```
>
>> t=FINDGEN(5,200)
>> x = t(1,*)
>> y = t(2,*)
>> z = t(3,*)
```

```
>>
>> p = PLOT3D(x, y, z,'o')
```

```
>
>
```

> I have found that function graphics are particularly picky about whether you pass them a column vector or row vector. Problems arise when you pass data in as a row vector (a 1xN array, as you are doing above). The solution is to reform your data into a column vector (Nx1 array), like so

```
>
> x = reform(t[1,*])
> y = reform(t[2,*])
> z = reform(t[3,*])
```

```
>
> p = plot3d(x, y, z, 'o')
```

Actually, visa versa, but we get the idea. :-)

x = t[1,*] is a column vector.

y = Reform(t[2,*]) is a row vector.

Cheers,

David

--

David Fanning, Ph.D.

Fanning Software Consulting, Inc.

Coyote's Guide to IDL Programming: <http://www.idlcoyote.com/>

Sepore ma de ni thue. ("Perhaps thou speakest truth.")

Subject: Re: PLOT3D format input

Posted by [Matthew Argall](#) on Wed, 18 Mar 2015 13:11:30 GMT

[View Forum Message](#) <> [Reply to Message](#)

Doh! That always gets me. A row vector is a vector full of columns and a column vector is vector full of rows ;-)

Subject: Re: PLOT3D format input

Posted by [Paul Van Delst\[1\]](#) on Wed, 18 Mar 2015 19:38:51 GMT

[View Forum Message](#) <> [Reply to Message](#)

Hello,

On 03/17/15 22:49, Matthew Argall wrote:

```
>> t=INDGEN(5,200)
```

```
>> x = t(1,*)
```

```
>> y = t(2,*)
```

```
>> z = t(3,*)
```

```
>>
```

```
>> p = PLOT3D(x, y, z,'o')
```

```
>
```

```
>
```

```
> I have found that function graphics are particularly picky about
```

```
> whether you pass them a column vector or row vector. Problems arise when
```

```
> you pass data in as a row vector (a 1xN array, as you are doing above).
```

```
> The solution is to reform your data into a column vector (Nx1 array),
```

```
> like so
```

```
>
```

```
> x = reform(t[1,*])
```

```
> y = reform(t[2,*])
```

```
> z = reform(t[3,*])
```

```
>
```

```
> p = plot3d(x, y, z, 'o')
```

Matthew's suggestion is a good one, but being the memory-layout-worrier that I am (preferring `t[*,0]` over `t[0,*]`), I would simply transpose the "t" array directly after reading it in from wherever it was created,

The original data...

```
IDL> t = findgen(5,200)
IDL> help, t
T          FLOAT    = Array[5, 200]
IDL> print, t[0,*]
0.00000
5.00000
10.0000
15.0000
20.0000
25.0000
30.0000
35.0000
40.0000
45.0000
.....
```

Transpose it for all subsequent use:

```
IDL> t = transpose(t)
IDL> help, t
T          FLOAT    = Array[200, 5]
IDL> print, t[* ,0]
0.00000  5.00000  10.0000  15.0000  20.0000
25.0000  30.0000  35.0000  40.0000  45.0000
....
```

When the trailing dimension is the degenerate one, IDL happily ignores it....

```
IDL> x = COS(t) * (1 + t / 10)
IDL> help, x
X          FLOAT    = Array[200, 5]
IDL> p = PLOT3D(x[* ,0], x[* ,1], x[* ,2], 'o')
```

cheers,

paulv

Subject: Re: PLOT3D format input
Posted by [lucsmm](#) on Thu, 19 Mar 2015 23:24:50 GMT
[View Forum Message](#) <> [Reply to Message](#)

On Wednesday, March 18, 2015 at 12:38:53 PM UTC-7, Paul van Delst wrote:

```

> Hello,
>
> On 03/17/15 22:49, Matthew Argall wrote:
>>> t=FINDGEN(5,200)
>>> x = t(1,*)
>>> y = t(2,*)
>>> z = t(3,*)
>>>
>>> p = PLOT3D(x, y, z,'o')
>>
>>
>> I have found that function graphics are particularly picky about
>> whether you pass them a column vector or row vector. Problems arise when
>> you pass data in as a row vector (a 1xN array, as you are doing above).
>> The solution is to reform your data into a column vector (Nx1 array),
>> like so
>>
>> x = reform(t[1,*])
>> y = reform(t[2,*])
>> z = reform(t[3,*])
>>
>> p = plot3d(x, y, z, 'o')
>
> Matthew's suggestion is a good one, but being the memory-layout-worrier
> that I am (preferring t[:,0] over t[0,*]), I would simply transpose the
> "t" array directly after reading it in from wherever it was created,
>
> The original data...
>
> IDL> t = findgen(5,200)
> IDL> help, t
> T          FLOAT      = Array[5, 200]
> IDL> print, t[0,*]
>    0.00000
>    5.00000
>   10.0000
>   15.0000
>   20.0000
>   25.0000
>   30.0000
>   35.0000
>   40.0000
>   45.0000
>   ....
>
> Transpose it for all subsequent use:
>
> IDL> t = transpose(t)

```

```
> IDL> help, t
> T          FLOAT    = Array[200, 5]
> IDL> print, t[* ,0]
>    0.00000    5.00000    10.0000    15.0000    20.0000
>    25.0000    30.0000    35.0000    40.0000    45.0000
>    ....
>
> When the trailing dimension is the degenerate one, IDL happily ignores
> it....
>
> IDL> x = COS(t) * (1 + t / 10)
> IDL> help, x
> X          FLOAT    = Array[200, 5]
> IDL> p = PLOT3D(x[* ,0], x[* ,1], x[* ,2], 'o')
>
>
> cheers,
>
> paulv
```

Thank you
Now I can plot it.
Sometimes I don't understand how this formatting works...

-LMM
