Subject: size(/dimen) that automatically fills in extra dimensions
Posted by Jeremy Bailin on Sun, 21 Jun 2015 03:38:04 GMT
View Forum Message <> Reply to Message

Before I write a quick routine that does this, it seems like someone must have done this already:

Does anyone have a drop-in replacement for SIZE(/DIMEN) that automatically fills in missing trailing dimensions with 1?

I.e. I have an array A that is always 3xN, but N could be 1, 2, or 3. I want to find out N, but

Size(A, /DIMEN)[1]

fails if N eq 1 because IDL drops the final dimension.

(even better: this would be a nice switch for the official SIZE function to have, if anyone is listening)

-Jeremy.

Subject: Re: size(/dimen) that automatically fills in extra dimensions
Posted by Dick Jackson on Sun, 21 Jun 2015 07:42:34 GMT
View Forum Message <> Reply to Message

On Saturday, 20 June 2015 20:38:07 UTC-7, Jeremy Bailin wrote:
> Before I write a quick routine that does this, it seems like someone must have done this already:
>
> Does anyone have a drop-in replacement for SIZE(/DIMEN) that automatically fills in missing trailing dimensions with 1?
>
> I.e. I have an array A that is always 3xN, but N could be 1, 2, or 3. I want to find out N, but
>
> Size(A, /DIMEN)[1]
>
> fails if N eq 1 because IDL drops the final dimension.
>
> (even better: this would be a nice switch for the official SIZE function to have, if anyone is listening)
>
> -Jeremy.

Hi Jeremy,

I've been in your shoes...

In case this is helpful, there is a way to force the array to have a (3, N) shape, using Reform:

```
IDL> a=indgen(5,1)
IDL> help,a
A               INT       = Array[5]     ; OK, the 1 has been dropped

IDL> a=reform(a,[5,1], /OVERWRITE)
IDL> help,a
A               INT       = Array[5, 1]

; Here's a handy routine when you want to ensure you have at least 'n' dimensions
;------
PRO EnsureNDims, x, nDims
IF Size(x, /N_Dimensions) GE nDims THEN RETURN
newDims = Replicate(1L, nDims)
newDims[0] = Size(x, /Dimensions) > 1 ; Will work even if x is scalar
x = Reform(x, newDims, /Overwrite)
END
;------
```

It can be interesting to see when this changes:

```
IDL> a=indgen([5,1])
IDL> help,a
A               INT       = Array[5]
IDL> ensurendims,a,2
IDL> help,a
A               INT       = Array[5, 1]
IDL> a=a
IDL> help,a
A               INT       = Array[5, 1]
; that was OK, didn't break it

IDL> b=a
IDL> help,b
B               INT       = Array[5]
; that broke it

IDL> a=b
IDL> help,a
A               INT       = Array[5]
; that broke 'a'

IDL> ensurendims,a,2
IDL> help,a
A               INT       = Array[5, 1]
IDL> a=a+1
IDL> help,a
A               INT       = Array[5]
```

```
; that broke it

IDL> a++
IDL> help,a
A               INT       = Array[5, 1]
; ... but that's OK!
```

Hope this helps!

Cheers,
-Dick

Dick Jackson Software Consulting Inc.
Victoria, BC, Canada --- http://www.d-jackson.com

Subject: Re: size(/dimen) that automatically fills in extra dimensions
Posted by Jeremy Bailin on Mon, 22 Jun 2015 13:43:27 GMT

View Forum Message <> Reply to Message

On Sunday, June 21, 2015 at 3:42:38 AM UTC-4, Dick Jackson wrote:
> On Saturday, 20 June 2015 20:38:07 UTC-7, Jeremy Bailin wrote:
>> Before I write a quick routine that does this, it seems like someone must have done this already:
>>
>> Does anyone have a drop-in replacement for SIZE(/DIMEN) that automatically fills in missing trailing dimensions with 1?
>>
>> I.e. I have an array A that is always 3xN, but N could be 1, 2, or 3. I want to find out N, but
>>
>> Size(A, /DIMEN)[1]
>>
>> fails if N eq 1 because IDL drops the final dimension.
>>
>> (even better: this would be a nice switch for the official SIZE function to have, if anyone is listening)
>>
>> -Jeremy.
>
> Hi Jeremy,
>
> I've been in your shoes...
>
> In case this is helpful, there is a way to force the array to have a (3, N) shape, using Reform:
>
> IDL> a=indgen(5,1)
> IDL> help,a
> A   INT   = Array[5]   ; OK, the 1 has been dropped

```
>
> IDL> a=reform(a,[5,1], /OVERWRITE)
> IDL> help,a
> A              INT       = Array[5, 1]
>
> ; Here's a handy routine when you want to ensure you have at least 'n' dimensions
> ;------
> PRO EnsureNDims, x, nDims
> IF Size(x, /N_Dimensions) GE nDims THEN RETURN
> newDims = Replicate(1L, nDims)
> newDims[0] = Size(x, /Dimensions) > 1 ; Will work even if x is scalar
> x = Reform(x, newDims, /Overwrite)
> END
> ;------
>
> It can be interesting to see when this changes:
>
> IDL> a=indgen([5,1])
> IDL> help,a
> A              INT       = Array[5]
> IDL> ensurendims,a,2
> IDL> help,a
> A              INT       = Array[5, 1]
> IDL> a=a
> IDL> help,a
> A              INT       = Array[5, 1]
> ; that was OK, didn't break it
>
> IDL> b=a
> IDL> help,b
> B              INT       = Array[5]
> ; that broke it
>
> IDL> a=b
> IDL> help,a
> A              INT       = Array[5]
> ; that broke 'a'
>
> IDL> ensurendims,a,2
> IDL> help,a
> A              INT       = Array[5, 1]
> IDL> a=a+1
> IDL> help,a
> A              INT       = Array[5]
> ; that broke it
>
> IDL> a++
> IDL> help,a
>  A               INT       = Array[5, 1]
```

> ; ... but that's OK!
>
> Hope this helps!
>
> Cheers,
> -Dick
>
> Dick Jackson Software Consulting Inc.
> Victoria, BC, Canada --- http://www.d-jackson.com

That's interesting... I can kind of see why certain ones do vs. don't, but I'm not sure I could have predicted each case a priori!

In this case, I don't actually need to change the dimensions, since the rest of my code works fine even when there's no trailing dimension -- I just need to be able to access its size. I could use this and then run Size right afterwards, but I've ended up writing it as a quick single function instead:

```
; Return's the length of the D-th dimension (starting with 1) of A,
; returning 1 for any missing trailing dimensions.
function size_d, a, d
  s = size(a, /dimen)
  if d le n_elements(s) then return, s[d]
  return, 1
end
```

-Jeremy.

Subject: Re: size(/dimen) that automatically fills in extra dimensions
Posted by Jeremy Bailin on Mon, 22 Jun 2015 13:59:46 GMT

View Forum Message <> Reply to Message

On Monday, June 22, 2015 at 9:43:33 AM UTC-4, Jeremy Bailin wrote:
> On Sunday, June 21, 2015 at 3:42:38 AM UTC-4, Dick Jackson wrote:
>> On Saturday, 20 June 2015 20:38:07 UTC-7, Jeremy Bailin wrote:
>>> Before I write a quick routine that does this, it seems like someone must have done this already:
>>>
>>> Does anyone have a drop-in replacement for SIZE(/DIMEN) that automatically fills in missing trailing dimensions with 1?
>>>
>>> I.e. I have an array A that is always 3xN, but N could be 1, 2, or 3. I want to find out N, but
>>>
>>> Size(A, /DIMEN)[1]
>>>
>>> fails if N eq 1 because IDL drops the final dimension.
>>>

>>> (even better: this would be a nice switch for the official SIZE function to have, if anyone is listening)
>>>
>>> -Jeremy.
>>
>> Hi Jeremy,
>>
>> I've been in your shoes...
>>
>> In case this is helpful, there is a way to force the array to have a (3, N) shape, using Reform:
>>

```
>>  IDL> a=indgen(5,1)
>>  IDL> help,a
>>  A               INT       = Array[5]     ; OK, the 1 has been dropped
>>
>>  IDL> a=reform(a,[5,1], /OVERWRITE)
>>  IDL> help,a
>>  A               INT       = Array[5, 1]
>>
>>  ; Here's a handy routine when you want to ensure you have at least 'n' dimensions
>>  ;------
>>  PRO EnsureNDims, x, nDims
>>  IF Size(x, /N_Dimensions) GE nDims THEN RETURN
>>  newDims = Replicate(1L, nDims)
>>  newDims[0] = Size(x, /Dimensions) > 1 ; Will work even if x is scalar
>>  x = Reform(x, newDims, /Overwrite)
>>  END
>>  ;------
>>
>>  It can be interesting to see when this changes:
>>
>>  IDL> a=indgen([5,1])
>>  IDL> help,a
>>  A               INT       = Array[5]
>>  IDL> ensurendims,a,2
>>  IDL> help,a
>>  A               INT       = Array[5, 1]
>>  IDL> a=a
>>  IDL> help,a
>>  A               INT       = Array[5, 1]
>>  ; that was OK, didn't break it
>>
>>  IDL> b=a
>>  IDL> help,b
>>  B               INT       = Array[5]
>>  ; that broke it
>>
>>  IDL> a=b
>>  IDL> help,a
>>  A               INT       = Array[5]
>>  ; that broke 'a'
>>
>>  IDL> ensurendims,a,2
>>  IDL> help,a
>>  A               INT       = Array[5, 1]
>>  IDL> a=a+1
>>  IDL> help,a
>>  A               INT       = Array[5]
>>  ; that broke it
>>
>>  IDL> a++
>>  IDL> help,a
>>  A               INT       = Array[5, 1]
>>  ; ... but that's OK!
>>
```

>> Hope this helps!
>>
>> Cheers,
>> -Dick
>>
>> Dick Jackson Software Consulting Inc.
>> Victoria, BC, Canada --- http://www.d-jackson.com

> That's interesting... I can kind of see why certain ones do vs. don't, but I'm not sure I could have predicted each case a priori!
>
> In this case, I don't actually need to change the dimensions, since the rest of my code works fine even when there's no trailing dimension -- I just need to be able to access its size. I could use this and then run Size right afterwards, but I've ended up writing it as a quick single function instead:

```
>
>  ; Return's the length of the D-th dimension (starting with 1) of A,
>  ; returning 1 for any missing trailing dimensions.
>  function size_d, a, d
>    s = size(a, /dimen)
>    if d le n_elements(s) then return, s[d]
>    return, 1
>  end
>
```

> -Jeremy.

Er, no that's not right -- that should be:

```
; Return's the length of the D-th dimension (starting with 0) of A,
; returning 1 for any missing trailing dimensions.
function size_d, a, d
```

```
  s = size(a, /dimen)
  if d lt n_elements(s) then return, s[d]
  return, 1
end
```

-Jeremy.