Subject: Assigning the values of a matrix to a larger matrix
Posted by Fabrice Lambert on Wed, 30 Apr 2014 20:52:42 GMT
View Forum Message <> Reply to Message

Hello,

Is there a way to write the following code in IDL/GDL without a for loop?

```
for i=0,90 do begin
   Base[*,*,i]=Input[*,*,0]
endfor
```

for example something like this (which unfortunately does not work:

```
Base[*,*,0:90]=Input[*,*,0]
```

Thanks,
Fabrice

Subject: Re: Assigning the values of a matrix to a larger matrix
Posted by Heinz Stege on Wed, 30 Apr 2014 21:21:35 GMT
View Forum Message <> Reply to Message

Hello Fabrice.

On Wed, 30 Apr 2014 13:52:42 -0700 (PDT), fabrice.lambert@gmail.com
wrote:

> Is there a way to write the following code in IDL/GDL without a for loop?
>
> for i=0,90 do begin
>     Base[*,*,i]=Input[*,*,0]
> endfor

Yes, you can write:

```
n=size(input,/dimensions)
base[0,0,0]=rebin(input,[n,91],/sample)
```

If  the base array does not have to be bigger than the 91x input
array, you don't need to create it explicitly and can more simply
write:

```
n=size(input,/dimensions)
base=rebin(input,[n,91],/sample)
```

Cheers, Heinz

Subject: Re: Assigning the values of a matrix to a larger matrix
Posted by Fabrice Lambert on Wed, 30 Apr 2014 23:05:11 GMT
View Forum Message <> Reply to Message

Hello Heinz,

That works, thanks a lot!

I find it very counter-intuitive, though: In my example to fill in the values we want in the 3rd dimension of the base vector, we have to create a 4th dimension where every index contains a copy of the original 3d-input vector.

Is there an easy way to understand that logic?

Thanks,
Fabrice

On Wednesday, April 30, 2014 5:21:35 PM UTC-4, Heinz Stege wrote:
> Hello Fabrice.
>
>
>
>
>
>
>> Is there a way to write the following code in IDL/GDL without a for loop?
>
>>
>
>> for i=0,90 do begin
>
>> Base[*,*,i]=Input[*,*,0]
>
>> endfor
>
>
>
> Yes, you can write:
>
>
>
> n=size(input,/dimensions)
>
> base[0,0,0]=rebin(input,[n,91],/sample)
>
>
>
> If the base array does not have to be bigger than the 91x input
>

> array, you don't need to create it explicitly and can more simply
>
> write:
>
>
>
> n=size(input,/dimensions)
>
> base=rebin(input,[n,91],/sample)
>
>
>
> Cheers, Heinz

---

Subject: Re: Assigning the values of a matrix to a larger matrix
Posted by Heinz Stege on Fri, 02 May 2014 18:16:45 GMT
View Forum Message <> Reply to Message

On Wed, 30 Apr 2014 16:05:11 -0700 (PDT), fabrice.lambert@gmail.com
wrote:

> Hello Heinz,
>
> That works, thanks a lot!
>
> I find it very counter-intuitive, though: In my example to fill in the values we want in the 3rd dimension of the base vector, we have to create a 4th dimension where every index contains a copy of the original 3d-input vector.
>
> Is there an easy way to understand that logic?
>
> Thanks,
> Fabrice
>
> On Wednesday, April 30, 2014 5:21:35 PM UTC-4, Heinz Stege wrote:
>> Hello Fabrice.
>>
>>
>>
>>
>>
>>
>>> Is there a way to write the following code in IDL/GDL without a for loop?
>>
>>>
>>
>>> for i=0,90 do begin

>>
>>>     Base[*,*,i]=Input[*,*,0]
>>
>>> endfor
>>
>>
>>
>>  Yes, you can write:
>>
>>
>>
>>  n=size(input,/dimensions)
>>
>>  base[0,0,0]=rebin(input,[n,91],/sample)
>>
>>
>>
>>  If  the base array does not have to be bigger than the 91x input
>>
>>  array, you don't need to create it explicitly and can more simply
>>
>>  write:
>>
>>
>>
>>  n=size(input,/dimensions)
>>
>>  base=rebin(input,[n,91],/sample)
>>
>>
>>
>>  Cheers, Heinz

Hello Fabrice,

I believe, something went absolutely wrong here. Are you really sure, that you get a correct result with my solution?

You say, that rebin(input,[n,91],/sample) has 4 dimensions. This means, that input itself has 3 dimensions (and the size of the 3rd dimension is greater or equal 2). It was my failure, not to think of this possibility. Something made me thinking, that it was an "extra dimension" you addressed with the zero in "Input[*,*,0]". Sorry for this.

I wonder, why IDL didn't throw an error message. Does your base array have more than 3 dimensions, which you don't want to touch with this

part of code?

Anyway, you want to write 3 dimensions into the base array. And as I understand now, your input array has 3 dimensions too, but you want to repeatedly use the first element of the third dimension. Here is a detailed example, how to do this without a loop:

```
input=indgen(5,4,2)
base=intarr(5,4,3)
base[0,0,0]=rebin(input[*,*,0],5,4,3,/sample)
```

A print displays the following result:

```
IDL> print,base
      0       1       2       3       4
      5       6       7       8       9
     10      11      12      13      14
     15      16      17      18      19

      0       1       2       3       4
      5       6       7       8       9
     10      11      12      13      14
     15      16      17      18      19

      0       1       2       3       4
      5       6       7       8       9
     10      11      12      13      14
     15      16      17      18      19
```

Another note: It is not mandatory, that for big arrays, this solution is faster than a loop. If your routine is time-consuming, you should test it.

I hope, this makes the things more clear. And sorry again for my mistake.

Cheers, Heinz

---

Subject: Re: Assigning the values of a matrix to a larger matrix
Posted by Fabrice Lambert on Fri, 02 May 2014 23:51:43 GMT
View Forum Message <> Reply to Message

Hello Heinz,

Indeed, your initial code did produce an error. What I meant with "that works" was that I read up on rebin and tested it on some simple examples.

You understood my problem well now. I have many variables with plenty of dimensions and I've

been looking for a way to avoid endless for loops to assign one dimension of one variable to another dimension of another variable. It works well enough with for loops, but I have a feeling there must be a more elegant solution.

I'll study your new solution, thanks a lot for the follow up.

-Fabrice

On Friday, May 2, 2014 2:16:45 PM UTC-4, Heinz Stege wrote:
> On Wed, 30 Apr 2014 16:05:11 -0700 (PDT),
>
> wrote:
>
>
>
>> Hello Heinz,
>
>>
>
>> That works, thanks a lot!
>
>>
>
>> I find it very counter-intuitive, though: In my example to fill in the values we want in the 3rd dimension of the base vector, we have to create a 4th dimension where every index contains a copy of the original 3d-input vector.
>
>>
>
>> Is there an easy way to understand that logic?
>
>>
>
>> Thanks,
>
>> Fabrice
>
>>
>
>> On Wednesday, April 30, 2014 5:21:35 PM UTC-4, Heinz Stege wrote:
>
>>> Hello Fabrice.
>
>>>
>
>>>
>
>>>

>
>>>
>
>>>
>
>>>
>
>>>> Is there a way to write the following code in IDL/GDL without a for loop?
>
>>>
>
>>>>
>
>>>
>
>>>> for i=0,90 do begin
>
>>>
>
>>>>     Base[*,*,i]=Input[*,*,0]
>
>>>
>
>>>> endfor
>
>>>
>
>>>
>
>>>
>
>>>  Yes, you can write:
>
>>>
>
>>>
>
>>>
>
>>>  n=size(input,/dimensions)
>
>>>
>
>>>  base[0,0,0]=rebin(input,[n,91],/sample)
>
>>>
>
>>>

>
>>>
>
>>> If the base array does not have to be bigger than the 91x input
>
>>>
>
>>> array, you don't need to create it explicitly and can more simply
>
>>>
>
>>> write:
>
>>>
>
>>>
>
>>>
>
>>> n=size(input,/dimensions)
>
>>>
>
>>> base=rebin(input,[n,91],/sample)
>
>>>
>
>>>
>
>>>
>
>>> Cheers, Heinz
>
>
>
>
>
> Hello Fabrice,
>
>
>
> I believe, something went absolutely wrong here. Are you really sure,
>
> that you get a correct result with my solution?
>
>
>
> You say, that rebin(input,[n,91],/sample) has 4 dimensions. This

> means, that input itself has 3 dimensions (and the size of the 3rd
> dimension is greater or equal 2). It was my failure, not to think of
> this possibility. Something made me thinking, that it was an "extra
> dimension" you addressed with the zero in "Input[*,*,0]". Sorry for
> this.
>
>
> I wonder, why IDL didn't throw an error message. Does your base array
> have more than 3 dimensions, which you don't want to touch with this
> part of code?
>
>
> Anyway, you want to write 3 dimensions into the base array. And as I
> understand now, your input array has 3 dimensions too, but you want to
> repeatedly use the first element of the third dimension. Here is a
> detailed example, how to do this without a loop:
>
>
> input=indgen(5,4,2)
> base=intarr(5,4,3)
> base[0,0,0]=rebin(input[*,*,0],5,4,3,/sample)
>
>
> A print displays the following result:
> IDL> print,base
>
>  0  1  2  3  4
>
>  5  6  7  8  9
>
> 10 11 12 13 14

>
>      15     16     17     18     19
>
>
>
>       0      1      2      3      4
>
>       5      6      7      8      9
>
>      10     11     12     13     14
>
>      15     16     17     18     19
>
>
>
>       0      1      2      3      4
>
>       5      6      7      8      9
>
>      10     11     12     13     14
>
>      15     16     17     18     19
>
>
>
> Another note: It is not mandatory, that for big arrays, this solution
>
> is faster than a loop. If your routine is time-consuming, you should
>
> test it.
>
>
>
> I hope, this makes the things more clear. And sorry again for my
>
> mistake.
>
>
>
> Cheers, Heinz

---

Subject: Re: Assigning the values of a matrix to a larger matrix
Posted by Fabrice Lambert on Wed, 07 May 2014 16:35:45 GMT
View Forum Message <> Reply to Message

Hello,

Thank you very much for the help with rebin. That works great to work on the last dimension. Unfortunately, rebin doesn't support changes to other dimensions. Example:

```
for i=0,127 do begin
   for j=0,63 do begin
      for t=0,11 do begin
         for b=0,5 do begin
            for p=0,1 do begin
               AtmConc[i,j,*,t,b,p]=SurfConc[i,j,t,b,p]*AtmFactor[i,j,*,t,b ,p]
            endfor
         endfor
      endfor
   endfor
endfor
```

Is there a way to avoid the for-loops in that case?

Thanks,
Fabrice

On Friday, May 2, 2014 2:16:45 PM UTC-4, Heinz Stege wrote:
> On Wed, 30 Apr 2014 16:05:11 -0700 (PDT),
>
> wrote:
>
>
>
>> Hello Heinz,
>
>>
>
>> That works, thanks a lot!
>
>>
>
>> I find it very counter-intuitive, though: In my example to fill in the values we want in the 3rd dimension of the base vector, we have to create a 4th dimension where every index contains a copy of the original 3d-input vector.
>
>>
>
>> Is there an easy way to understand that logic?
>
>>
>
>> Thanks,
>
>> Fabrice

>
>>
>
>> On Wednesday, April 30, 2014 5:21:35 PM UTC-4, Heinz Stege wrote:
>
>>> Hello Fabrice.
>
>>>
>
>>>
>
>>>
>
>>>
>
>>>
>
>>>
>
>>>
>
>>>> Is there a way to write the following code in IDL/GDL without a for loop?
>
>>>
>
>>>>
>
>>>
>
>>>> for i=0,90 do begin
>
>>>
>
>>>>     Base[*,*,i]=Input[*,*,0]
>
>>>
>
>>>> endfor
>
>>>
>
>>>
>
>>>
>
>>> Yes, you can write:
>
>>>
>
>>>
>

>
>>>
>
>>> n=size(input,/dimensions)
>
>>>
>
>>> base[0,0,0]=rebin(input,[n,91],/sample)
>
>>>
>
>>>
>
>>>
>
>>> If the base array does not have to be bigger than the 91x input
>
>>>
>
>>> array, you don't need to create it explicitly and can more simply
>
>>>
>
>>> write:
>
>>>
>
>>>
>
>>>
>
>>> n=size(input,/dimensions)
>
>>>
>
>>> base=rebin(input,[n,91],/sample)
>
>>>
>
>>>
>
>>>
>
>>> Cheers, Heinz
>
>
>
>

> Hello Fabrice,
>
>
> I believe, something went absolutely wrong here. Are you really sure,
> that you get a correct result with my solution?
>
>
> You say, that rebin(input,[n,91],/sample) has 4 dimensions. This
> means, that input itself has 3 dimensions (and the size of the 3rd
> dimension is greater or equal 2). It was my failure, not to think of
> this possibility. Something made me thinking, that it was an "extra
> dimension" you addressed with the zero in "Input[*,*,0]". Sorry for
> this.
>
>
> I wonder, why IDL didn't throw an error message. Does your base array
> have more than 3 dimensions, which you don't want to touch with this
> part of code?
>
>
> Anyway, you want to write 3 dimensions into the base array. And as I
> understand now, your input array has 3 dimensions too, but you want to
> repeatedly use the first element of the third dimension. Here is a
> detailed example, how to do this without a loop:
>
>
> input=indgen(5,4,2)
> base=intarr(5,4,3)
> base[0,0,0]=rebin(input[*,*,0],5,4,3,/sample)

> A print displays the following result:
>
> IDL> print,base
>
>  0  1  2  3  4
>
>  5  6  7  8  9
>
>  10  11  12  13  14
>
>  15  16  17  18  19
>
>  0  1  2  3  4
>
>  5  6  7  8  9
>
>  10  11  12  13  14
>
>  15  16  17  18  19
>
>  0  1  2  3  4
>
>  5  6  7  8  9
>
>  10  11  12  13  14
>
>  15  16  17  18  19
>
> Another note: It is not mandatory, that for big arrays, this solution
>
> is faster than a loop. If your routine is time-consuming, you should
>
> test it.
>
> I hope, this makes the things more clear. And sorry again for my
>
> mistake.

>
>
>
> Cheers, Heinz

---

Subject: Re: Assigning the values of a matrix to a larger matrix
Posted by Heinz Stege on Wed, 07 May 2014 17:51:19 GMT
View Forum Message <> Reply to Message

On Wed, 7 May 2014 09:35:45 -0700 (PDT), Fabrice Lambert wrote:

> Is there a way to avoid the for-loops in that case?

Yes again. REBIN has a partner. It's name is REFORM. REFORM is able to
add extra dimensions within an array. Example
  a=indgen(3,2)
  a=reform(a,3,1,2,/overwrite)
After this REBIN can be used to increase the new dimension.

For more information you may want to read
http://www.idlcoyote.com/tips/rebin_magic.html.

Cheers, Heinz

---

Subject: Re: Assigning the values of a matrix to a larger matrix
Posted by Fabrice Lambert on Wed, 07 May 2014 23:40:57 GMT
View Forum Message <> Reply to Message

Just what I needed.
Thanks a lot Heinz!

-Fabrice

On Wednesday, May 7, 2014 1:51:19 PM UTC-4, Heinz Stege wrote:
> On Wed, 7 May 2014 09:35:45 -0700 (PDT), Fabrice Lambert wrote:
>
>
>
>> Is there a way to avoid the for-loops in that case?
>
>
>
> Yes again. REBIN has a partner. It's name is REFORM. REFORM is able to
>
> add extra dimensions within an array. Example

>
>  a=indgen(3,2)
>
>  a=reform(a,3,1,2,/overwrite)
>
> After this REBIN can be used to increase the new dimension.
>
>
>
> For more information you may want to read
>
> http://www.idlcoyote.com/tips/rebin_magic.html.
>
>
>
> Cheers, Heinz

---